Preserving Data Secrecy in Inter-organizational Process Mining

Valerio Goretti[0000−0001−9714−4278], Davide Basile[0000−1111−2222−3333], Luca Barbaro[0000−0002−2975−5330], and Claudio Di Ciccio[0000−0001−5570−0475]

Sapienza University of Rome, Italy
name.surname@uniroma1.it

Abstract. Inter-organizational business processes involve multiple independent organizations collaborating to achieve mutual interests. Process mining techniques have the potential to allow these organizations to enhance operational efficiency, improve performance, and deepen the understanding of their business based on the recorded process event data. However, inter-organizational process mining faces substantial challenges, including topical secrecy concerns: The involved organizations may not be willing to expose their own data to run mining algorithms jointly with their counterparts or third parties. In this paper, we introduce a novel approach that unlocks process mining on multiple actors' process event data while safeguarding the secrecy and integrity of the original records in an inter-organizational business setting. To ensure that the data acquisition, merging and elaboration phases are secure and that the processed information is hidden from involved and external actors alike, our approach resorts to decentralized trusted applications running in Trusted Execution Environments (TEEs). We show the feasibility of our solution by showcasing its application to a healthcare scenario.

Keywords: Collaborative Business Processes · Trusted Execution Environment · Encryption · Confidential Computing

1 Introduction

In today's business landscape, organizations constantly seek ways to enhance operational efficiency, increase performance, and gain valuable insights to improve their processes. Process mining offers techniques to discover, monitor, and improve business processes by extracting knowledge from chronological records known as *event logs*. Organizations record in these ledgers events referring to activities and interactions occurring within a business process. The vast majority of process mining contributions consider *intra-organizational* settings, in which business processes are executed inside individual organizations. However, organizations increasingly recognize the value of collaboration and synergy in achieving operational excellence. *Inter-organizational* business processes involve several independent organizations actively cooperating to achieve a shared objective. Despite the advantages in terms of transparency, performance optimization, and benchmarking that companies can gain from such practices, inter-organizational process mining raises challenges that make it still hardly applicable. The major issue concerns confidentiality. Companies are reluctant to outsource to their partners inside information that is required to execute process mining algorithms. Indeed, the sharing of sensitive operational data across organizational boundaries introduces concerns about data privacy, security, and compliance with regulations. *Trusted Execution Environments* (TEEs) can serve as fundamental enablers to balance the need for insights with the imperative to protect sensitive information in inter-organizational settings. TEEs offer secure contexts that guarantee code integrity and data confidentiality in external devices. *Trusted applications* are tamper-proof software objects running in these environments.

In this paper, we propose a novel approach for inter-organizational process mining that resorts to trusted applications to preserve the secrecy and integrity of shared data. To pursue this aim, we design a decentralized software architecture for a four-staged procedure: (i) the initial exchange of preliminary metadata,(ii) the attestation of the miner entity, (iii) the secure transmission of encrypted data amid multiple parties, (iiii) the privacy-preserving merge of the shared information segments followed by the isolated and verifiable computation of process discovery algorithms on joined data. We evaluate our proof-of-concept implementation against synthetic and real-world-based data with a convergence test and memory effectiveness assessment.

The remainder of the paper is structured as follows: Sect. 2 provides an overview of related work inherent to the theme of inter-organizational process mining. In Sect. 3, we introduce a use case example that considers a healthcare scenario. The high-level architecture of our solution is presented in Sect. 4. Following on from this, we instantiate the addressed design principles in Sect. 5 focusing on the employed technologies, workflow, and implementation. In Sect. 6, we discuss our solution. Finally, we conclude and present directions for future work in Sect. 7.

2 Related Work

While inter-organizational process mining remains a consistent challenge, the academic literature has introduced a limited set of solutions. In the subsequent section, we enumerate these contributions, highlighting both their commonalities and distinctions in comparison to our work. The work of Müller et al. [12] pays attention to data privacy and security within third-party systems that mine data generated from external providers on demand. To safeguard the integrity of data earmarked for mining purposes, their research introduces a conceptual architecture that entails the execution of process mining algorithms within a cloud service environment, fortified with trusted execution environments. Drawing inspiration from this foundational contribution, our research work endeavors to design a decentralized approach characterized by organizational autonomy in the execution of process mining algorithms, devoid of synchronization mechanisms involvement taking place between the involved parties. A notable departure from the Müller et al. framework lies in the fact that, in our architectural design, each participating organization retains the discretion to choose when and how mining operations are conducted. Moreover, we bypass the idea of fixed roles, engineering a peer-to-peer scenario in which organizations can simultaneously be data provisioners or miners. Elkoumy et al. [6,5] present a framework called Shareprom, which, like our work, offers a means for independent entities to execute process mining algorithms in inter-organizational settings while safeguarding their proprietary input data from exposure to external parties operating within the same context. Shareprom's functionality is confined to the execution of operations involving event log abstractions [1] represented as directed acyclic graphs, which the parties employ as intermediate pre-elaboration to be fed into secure multiparty computation (SMPC) [3] sessions. In contrast to our approach, where the exchanged data consists of encrypted source logs, the reliance of Shareprom on this specific graph representation imposes constraints that may prove limiting in various process mining scenarios, as stated by the authors. Given that process mining encompasses a wide array of data types and representations, we acknowledge the potential need for alternative data structures in diverse process mining contexts. Moreover, SMPC-based solutions require computationally intensive operations and synchronous cooperation among multiple parties, which make these protocols challenging to manage as the number of participants scales up [13]. In our research work, the secure computation is contained within single elaborators and does not require constant communication with external parties once the input data is exchanged. In the course of our research endeavor, we are confronted with the imperative task of integrating event logs originating from different data sources and constructing coherent traces that describe collaborative process instances. Consequently, we engage in a comprehensive examination of various methodologies delineated within the literature, each of which offers insights into the merging of event logs within inter-organizational settings. Among the array of potential solutions in this domain, the work of Claes et al. [2] holds particular significance for our research efforts. This seminal study introduces a two-step mechanism operating at the structured data level, contingent upon the configuration and subsequent application of merging rules. Each such rule delineates the criteria, namely the relations between attributes of the traces and/or the activities, that two distinct traces must satisfy in order to be combined. In contrast, the research by Hernandez et al. [9] posits a methodology functioning at the raw data level. This approach represents traces and activities as *bags-of-words* vectors, subject to cosine similarity measurements to discern links and relationships between the traces earmarked for combination. An appealing aspect of this approach lies in its capacity to generalize the challenge of merging without necessitating a priori knowledge of the underlying semantics inherent to the logs under consideration. However, we have diverged from adopting this particular approach due to considerations inherent to computational overhead. This substantial computational load carries the potential to impact both the scalability and performance of our solution.

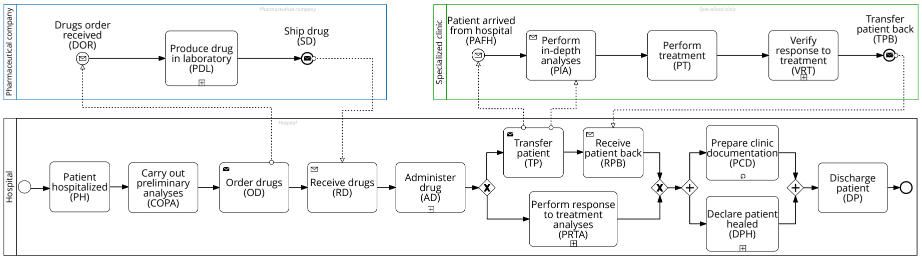

Pharmaceutical company
Drugs order received (DOR)
Produce drug in laboratory (PDL)
Ship drug (SD)
Specialized clinic
Patient arrived from hospital (PAFH)
Perform in-depth analyses (PIA)
Perform treatment (PT)
Verify response to treatment (VRT)
Transfer patient back (TPB)
Hospital
Patient hospitalized (PH)
Carry out preliminary analyses (COPA)
Order drugs (OD)
Receive drugs (RD)
Administer drug (AD)
Transfer patient (TP)
Receive patient back (RPB)
Perform response to treatment analyses (PRTA)
Prepare clinic documentation (PCD)
Declare patient healed (DPH)
Discharge patient (DP)

Fig. 1: A BPMN collaboration diagram of the healthcare scenario.

Table 1: Cases 312 and 711 recorded in the event logs of the Hospital, the Specialized clinic, and the Pharmaceutical company.

Hospital		
Case	Timestamp	Activity
312	2022-07-14T10:36	PH
312	2022-07-14T16:36	COPA
711	2022-07-14T17:21	PH
312	2022-07-14T17:36	OD
711	2022-07-14T23:21	COPA
711	2022-07-15T00:21	OD
711	2022-07-15T18:55	RD
312	2022-07-15T19:06	RD
711	2022-07-15T20:55	AD
312	2022-07-15T21:06	AD
312	2022-07-15T22:06	TP
711	2022-07-16T00:55	PRTA
711	2022-07-16T00:55	PCD
711	2022-07-16T02:55	DPH
711	2022-07-16T04:55	DP
312	2022-07-16T07:06	RPB
312	2022-07-16T09:06	DPH
312	2022-07-16T09:06	PCD
312	2022-07-16T11:06	DP

Pharmaceutical company		
Case	Timestamp	Activity
312	2022-07-15T09:06	DOR
711	2022-07-15T09:30	DOR
312	2022-07-15T11:06	PDL
711	2022-07-15T11:30	PDL
312	2022-07-15T13:06	SD
711	2022-07-15T13:30	SD

Specialized clinic		
Case	Timestamp	Activity
312	2022-07-16T00:06	PAFH
312	2022-07-16T01:06	PIA
312	2022-07-16T03:06	PT
312	2022-07-16T04:06	VRT
312	2022-07-16T05:06	TPB

T_{312} =⟨ PH, COPA, OD, DOR, PDL, SD, RD, AD, TP, PAFH, PIA, PT, VRT, TPB, RPB, DPH, PCD, DP ⟩
T_{711} =⟨ PH, COPA, OD, DOR, PDL, SD, RD, AD, TP, PAFH, DPH, PCD, DP ⟩

3 Motivating Scenario

For our motivating scenario, we focus on a simplified hospitalization process for the treatment of rare diseases that involves the cooperation of three parties: the Hospital, the Pharmaceutical organization, and the Specialized clinic. The process scheme is depicted in the BPMN diagram shown in Fig. 1. For the sake of simplicity, we describe the process through two cases. Alice's journey (case 312) begins when she enters the hospital for the preliminary examinations (the *patient hospitalized* event, PH). The Hospital then places to the Pharmaceutical company an order for the drugs (OD) needed to treat Alice's specific condition. Afterwards, the Pharmaceutical company acknowledges that the drugs order is received (DOR), proceeds to produce the drugs in the laboratory (PDL), and ships the drugs (SD) back to the Hospital. Upon receiving the medications, the Hospital administer the drug (AD), and conducts an assessment to determine if Alice can be treated internally. If specialized care is required, Alice is moved from the Hospital to the Specialized clinic (PAFH). When the patient arrives from the Hospital (PAFH), the Specialized clinic performs in-depth analyses (PIA) and proceeds with the treatment (PT). Once the Specialized clinic had completed the evaluations and verified the response to the alternative treatment (VRT), it transfers the patient back TPB. The Hospital receive the Alice patient back (RPB) and prepares the necessary clinic documentation (PCD). If Alice has successfully recovered, declares her as healed (DPH). When Alice's treatment is complete, the Hospital discharges the patient (DP). Bob enters the Hospital a few hours later than Alice. His hospitalization process is similar to Alice's. However, he does not need specialized care, and his case (711) is only treated by the Hospital. Therefore, the Hospital perform the response to treatment analyses (PRTA) instead of transferring him to the Specialized clinic. Both the National Institute of Statistics of the country in which the three organizations reside, together with the University that hosts/manages the hospital, wish to uncover information on this inter-organizational process for reporting and auditing purposes [**?**] via process analytics. The involved organizations share the urge for such an analysis, and wish to be able to repeat the mining task also in-house. The Hospital, the Specialized clinic, and the Pharmaceutical company have a partial view of the overall unfolding of the inter-organizational process as they record the events stemming from the parts of their pertinence. In Table 1, e.g., we show the traces 312 and 711 recorded by the Hospital (i.e., T^H_{312} and T^H_{711}), the Specialized clinic (i.e., T^S_{312} and T^S_{711}), and the Pharmaceutical company (i.e., T^C_{312} and T^C_{711}). Those traces are projections of the two combined ones for the whole inter-organizational process: T_{312} =⟨PH, COPA, OD, DOR, PDL, SD, RD, AD, TP, PAFH, PIA, PT, VRT, TPB, RPB, DPH, PCD, DP⟩ and T_{711} =⟨PH, COPA, OD, DOR, PDL, SD, RD, AD, TP, PAFH, DPH, PCD, DP⟩. Results stemming from the analysis of the local traces would not provide a full picture. Data should be merged. However, to preserve the privacy of the people involved and safeguard the confidentiality of the information, the involved parties cannot give open access to their traces to other organizations. The diverging interests (being able to conduct process mining on data from multiple sources without giving away the local event logs in-clear) motivate our research. In the following, we describe the design of our solution.

4 Design

In this section, we present the high-level architecture underlying our solution. We consider the main functionalities of each component, avoiding details on the employed technologies discussed in the next sections. Once we introduced the architecture, we focus on the `Secure Miner` component that represents the core of our contribution.

4.1 Architecture at large

Our architecture involves different organizational ecosystems characterized by one or more machines. An `Organization` may assume one of the following two different roles or both: *provisioner* if it delivers local event logs to be collaboratively mined; a *miner* whenever it applies process mining algorithms using local event logs retrieved from provisioners. Provisioner `Organizations` collaborate to achieve common objectives and compose inter-organizational business processes whose event logs are scattered across multiple places. Each provisioner produces event logs, recording the operations executed to complete its part in the inter-organizational business process. In Fig. 2, we propose the

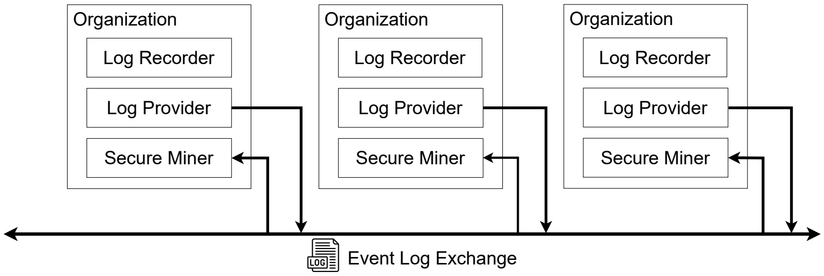


Fig. 2: High-level architectural overview.

high-level schematization of our solution. Organizations embed three main components, which we describe next: the Log Recorder, the Log Provider, and the Secure Miner. The maintenance of event logs is the core task performed by the Log Recorder. This component registers the events taking place in provisioner Organizations. The Hospital and the other parties in our running example record Alice and Bob's traces using their Log Recorders. The Log Recorder is queried by local Log Providers of the same Organization for event logs to be fed into remote Secure Miners. The Log Provider component delivers on-demand data to Secure Miners. It controls access to local event logs by authenticating data requests generated by miners. Log Providers reject demands from unauthorized parties and only permit Secure Miners to use the data. In our motivating scenario, the Specialized clinic, Pharmaceutical company, and the Hospital leverage Log Providers to authenticate the miner party before sending their logs. The Secure Miner shelters external event logs inside a miner ecosystem by preserving data confidentiality and integrity. We provide an in-depth focus on the Secure Miner as follows.

4.2 Secure Miner

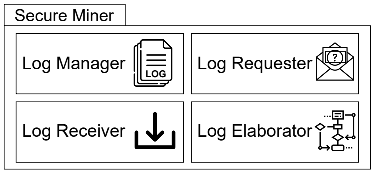


Fig. 3: Subcomponents of the Secure Miner.

The primary objective of the Secure Miner is to allow miners to securely execute process mining algorithms using event logs retrieved from provisioners such as the Specialized clinic, Pharmaceutical company, and the Hospital of our running example. Secure Miners are isolated components that guarantee tamper-proofing and data confidentiality. In Fig. 3, we show a schematization of a Secure Miner in which we distinguish four different subcomponents: the Log Manager, the Log Requester, the Log Receiver, and the Log Elaborator. Event logs belonging to provisioners are locked in the Secure Miner. We handle these data via the Log Manager which prevents malicious parties from having direct access to event logs. These unauthorized entities include any component of the miner Organization outside the Secure Miner. Referring to our motivating scenario, the Log Manager of the miner isolates the traces of Alice and Bob from secrecy-attempting actions generated outside the Secure Miner. The Log Requester and the Log Receiver are the subcomponents that we employ during the event log exchange. Log Requesters send authenticable data requests to the Log Provider component of provisioners. The Log Receiver collects event logs sent by Log Providers and entrusts them to the Log Manager. The miner of our motivating scenario employs these two components to retrieve the traces of Alice and Bob from the provisioners. and to collect this information in the Secure Miner. The Log Elaborator provides the functionality to securely execute process mining algorithms inside the Secure Miner. When activated, the Log Elaborator merge the traces locked in the Secure Miner in order to have a global view on the inter-organizational process comprensive of activities executed by each the party involved. Aggregated data is employeed by the Log Elaborator as input of process mining procedures. Mentioning our motivating scenario, the Log Elaborator combine the traces referring to the cases of Alice (i.e., T^{H}_{312}, T^{S}_{312}, and T^{C}_{312}) and Bob (i.e, T^{H}_{711}, T^{S}_{711}, and T^{C}_{711}) generating the chronologically sorted traces T_{312} and T_{711} to be fed into mining algorithms.

5 Realization

In this section, we outline the technical aspects concerning the realization of our approach. Therefore we first present the enabler technologies through which we instantiate the design principles presented in Sect. 4. After that, we discuss the interaction workflow between the instantiated technologies. Finally, we show the implementation details.

5.1 Deployment

As follows, we bridge the gap between high-level system architecture and its practical realization. Fig. 4 depicts a *UML deployment diagram* [10] that aims to help with understanding the instantiated infrastructure.

In our solution, we make a differentiation between the technologies designated for mining, denoted as Miner Machines, and those specifically associated with provisioners, identified as Provisioner Machines. To enhance clarity, we maintain the separation of these *devices* in the accompanying diagram. However, organizations have the flexibility to opt for integrated technologies that incorporate both mining and provisioning functionalities. Using our motivating scenario as an example, the Hospital can be equipped with machines aimed for both mining and provisioning, while the Specialized clinic can make use of separate devices. We included the Log Recorder, the Log Provider, and Secure Miner (already discussed in Sect. 4) as abstract *components* of the diagram, whose manifestation are described as follow.

Provisioner Machines encompasses Log Recorders and Log Providers incorporating their core functionalities aimed at generating and transmitting event logs. Within the organizational context, we manifest the Log Recorder in the

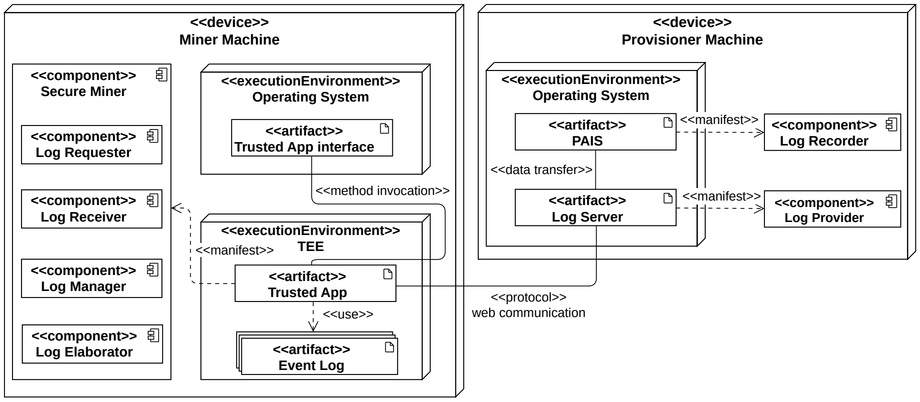


Fig. 4: UML deployment diagram.

Process Aware Information System (PAIS), which plays a crucial role in managing various business processes, including accounting and resource management [4]. In our motivating scenario, the Hospital and the other provisioner parties generate the traces Alice and Bob through this type of systems. In our solution, the PAIS grants access to the Log Server, enabling it to retrieve event logs. The Log Server, on the other hand, embody the functionalities of the Log Provider, implementing web services aimed at handling remote data requests and providing event log data to miners. The Hospital, the Specialized Clinic and the Pharmaceutical Company of our running example employes Log Servers adhering to established web standards such as HTTP[1], FTP[2], and Goopher[3]. Both the PAIS and Log Server run on the top the Operating System of the Provisioner Machine.

The Miner Machine is characterized by two distinct *execution environments*: the Operating System and the Trusted Execution Environment (TEE). TEEs establish an isolated context separate from the normal Operating System, safeguarding code and data through hardware-based encryption mechanisms. This technology relies on specialized components of the Miner Machine's CPU capable of managing encrypted data within a reserved section of RAM [?]. We leverage the security guarantees provided by TEEs to isolate a Trusted App responsible for fulfilling the functions of the Secure Miner and its associated subcomponents. The Trusted App consolidates the logic required for generating verifiable data requests, receiving external event logs, securely storing them within the TEE, and executing process mining algorithms. All procedures executed by the Trusted App are tamper-proof. The TEE ensures the integrity of the Trusted App code, protecting it against malicious manipulations and unauthorized access by entities operating within the Operating System. Additionally, we utilize the

[1] https://www.w3.org/Protocols/rfc2616/rfc2616.html
[2] https://www.w3.org/Protocols/rfc959/
[3] https://datatracker.ietf.org/doc/html/rfc1436

isolated environment of TEEs to securely store event log data (e.g., the traces of Alice and Bob of our example) from provisioner organizations within the Miner Machine. The TEE safeguards this sensitive information alongside a unique asymetric key couple used for attestation purposes (i.e., public and private keys), preventing exposure to the Operating System. Access to data located in the TEE is restricted solely to the Trusted App. Users interact with the Trusted App through the Trusted App Interface, which serves as the exclusive communication channel. The Trusted App offers secure methods, invoked by the Trusted App Interface, for safely receiving information from the Operating System and outsourcing the results of computations, maintaining a high level of data security.

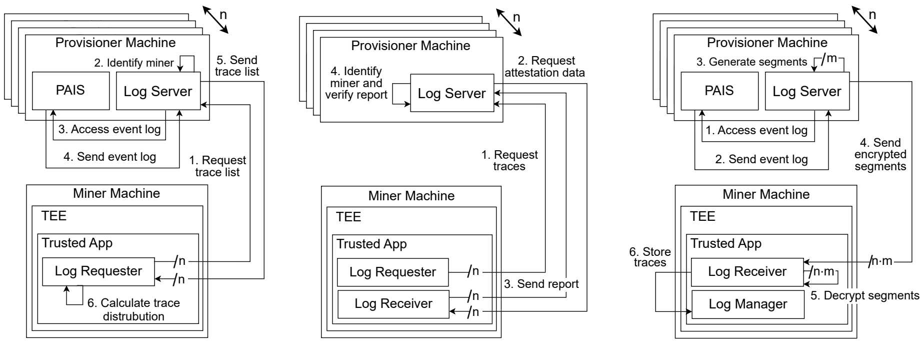
In the next subsection, we elucidate the interaction between the newly introduced technologies.

5.2 Workflow

We separate the workflow into subsequent processes, namely *initialization*, *remote attestation*, *data transmission*, and *computation*. This sequence of activities is enacted by two principal entities: a Miner Machine and a variable number, denoted in as n, of Provisioner Machines. We depict each stage separately in Fig. 5 and Fig. 6.

Initialization. The objective of the initialization stage is to inform the miner about the distribution of traces related to a business process among the Provisioner Machines, before the commencement of the computation. At the onset of this process, the Log Requester component within the Trusted App issues n requests to the Log Server components of the provisioners for the list of owned traces (1, in Fig. 5(a)). Following sender authentication (2), each Log Server retrieves the local event log from the PAIS (3, 4) and subsequently responds to the Log Requester by providing a list of its associated traces (5). After collecting these n responses, the Log Requester delineates the distribution of traces. In the context of our motivating scenario, by the conclusion of the initialization phase, the miner gains knowledge that traces belonging Bob's process, specifically T^{H}_{711} and T^{C}_{711}, are exclusively retained by the Hospital and the Specialized Clinic. In contrast, traces recording to Alice's process, denoted as T^{H}_{312}, T^{C}_{312}, and T^{S}_{312}, are scattered across all three organizations.

Remote Attestation. The remote attestation serves the purpose of establishing trust between miners and provisioners in the context of fulfilling data requests. This procedure has a dual objective: (i) to furnish provisioners with compelling evidence that the data request for an event log originates from a Trusted App currently executing within a TEE (Trusted Execution Environment), and (ii) to confirm the precise nature of the Trusted App as an authentic Secure Miner software entity. Upon the initiation of a new log request by the Log Requester (1, in Fig. 5(b)), each of the n Log Servers commences the verification process by requesting the necessary information from the Log Receiver to conduct the attestation (2). Subsequently, the Log Receiver generates a report containing the measurement of the Trusted App, which is defined as the hash value of



(a) Initialization (b) Remote Attestation (c) Data Transmission

Fig. 5: Schematization of the initialization, remote attestation and data transmission phase.

the combination of the code and initial data of the software. Once this report is signed using the attestation private key associated with the hardware of the Miner Machine, it is transmitted from the Log Receiver to the Log Servers (3). The Log Servers authenticate the miner and decrypt the report using its attestation public key (4). In this last step, the Log Servers undertake a comparison procedure in which they juxtapose the measurement found within the decrypted report against a predefined reference value associated with the source code of the Secure Miner.

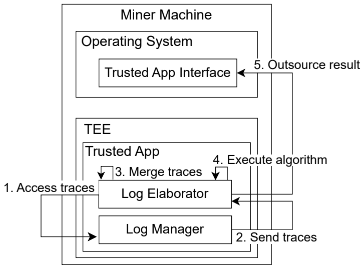


Fig. 6: Schematization of the computation phase.

Data Transmission. Once verified the trusted nature of the Trusted App, the Log Servers can proceed with the transmission of their traces. To accomplish this, each Log Server retrieves the event log from the PAIS (1 and 2, in Fig. 5(c)), and divides it into segments, resulting in m segments (3). Each of these segments contains a variable number of complete traces, with the cumulative size remaining within the threshold specified by the miner as a parameter of the initial request. As an illustrative example from our motivating scenario, the Log Server of the Hospital may structure the segmentation such that T^H_{312} and T^H_{711} reside within the same segment, whereas the Specialized clinic might have T^S_{312} and T^S_{711} in separate segments. Subsequently, the n Log Servers transmit their m encrypted segments to the Log Receiver of the Trusted App (4). The Log Receiver, in turn, decrypts each of the n • m responses (5) and securely stores the traces within the TEE through the Log Manager (6).

Merging and Computation. To start a computation routine, the Trusted App require all the provisioners to have delivered traces referring to the same process instances. For example, when T^H_{312}, T^S_{312} and T^C_{312} have all been received by the Trusted App, the process instance associated with Alice becomes eligible for computation. Upon meeting this condition, the Log Elaborator component requests the traces earmarked for computation from the Log Manager (1 and 2, in Fig. 6). To reconstruct a complete process instance, traces belonging to the same process instance must be merged by the Log Elaborator in a single trace (e.g., T_{312} for Alice case) comprensivhe of all the events in the partial traces (e.g., T^{H}_{312}, T^{S}_{312} and T^{C}_{312} for Alice case). To accomplish this, the Log Elaborator applies a specific *merging schema* as stated in [2]. According to this research work, a merging schema is defined as a rule specifying the attributes that link two traces during the merging process. In our illustrative scenario, the traces of Alice and Bob are accomunated by the same case id (i.e, 312 for Alice case, and 711 for Bob case). Therefore, the merging schema to combine their traces is contingent upon the linkage established by the concept:name attribute, as exemplified below:

```
merge all traces where
  "concept:name" of trace in log A
        equals
  "concept:name" of trace in log B
```

We underline that our solution allows for various merging schemas involving different trace attributes. After the merging operation (3), the Log Elaborator proceeds to input the merged traces into the process mining algorithm (4). Ultimately, the outcome of the computation is relayed by the Log Elaborator from the TEE to the Trusted App Interface running atop the Operating System of the Miner Machine (5).

5.3 Implementation

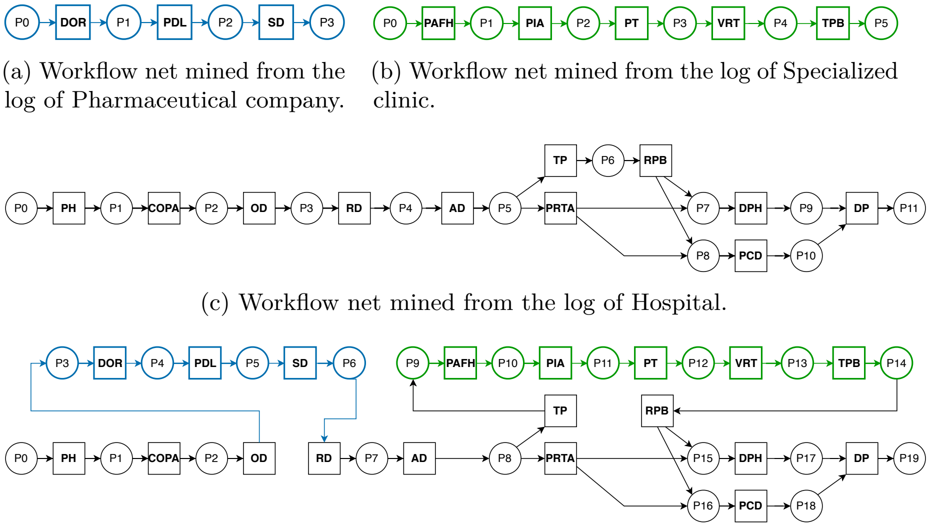
Punti da toccare: link git, intel sgx, EGo in golang, comunicazione trusted app - log server in HTTP, Heuristic Miner come algoritmo prof of concept Process Mining che genera Workflownet. Più coincisi il possibile.

The proposed implementation integrates a trusted application within a secure execution environment, complemented by the inclusion of event logs to address the issue outlined in the motivating scenario. The source code is accessible at the following URL: https://github.com/dave0909/TEExProcessMining/.

To realize the implementation of trusted applications, we employed EGo,[4] a framework designed for encoding trusted application in the Go[5] programming language. Contained within the trusted application is the Secure Miner module, which facilitates the requisition, administration, and processing of logs from external organizations. To exemplify our approach's capability for conducting process analysis, we implemented a renowned process discovery algorithm within the Secure Miner.

[4] https://www.edgeless.systems/products/ego/

[5] https://go.dev



(a) Workflow net mined from the log of Pharmaceutical company.

(b) Workflow net mined from the log of Specialized clinic.

(c) Workflow net mined from the log of Hospital.

(d) Workflow net mined from the merged log in inter-organizational setting.

Fig. 7: Outputs used for the convergence test.

6 Discussion

In this section, we evaluate the proposed approach. The primary objective of the Sect. 6.2 is to delve into a convergence analysis by evaluating the efficacy of the collaborative data exchange process. We then took into assessment the memory usage by measuring the RAM usage in diverse parameter configurations in **??**. In **??**, we validate the addressed approach using real-world data to conclude the discussion.

6.1 Datasets

6.2 Convergence

We take into analysis the convergence of the process discovery outputs, as a way of validating the correct operation of the event log exchange mechanism, . Specifically, we generated the workflow nets computed in an intra-organizational setting, in which each organization directly mines its own event log. Subsequently, we employed our approach with a miner actor that computes the same discovery algorithm using an inter-organizational event log obtained as a result of the log exchange and the merging mechanisms. To run the test, we used the synthetic event logs devised from our motivating scenario whose BPMN is depicted in Fig. 1. The size of the `Hospital`, `Specialized clinic`, and `Pharmaceutical company` event logs are 4.8 MB, 1.1 MB, and 1.6 MB respectively. Each log contains the standard value of 1000 traces, in accordance with the Sepsis Cases [11] event log. Upon visual examination of Fig. 7, we observe that the workflow net computed through our approach, displayed in Fig. 7(d), encapsulates the structure and behavior of the workflow nets derived from the intra-organizational discovery procedures depicted in Fig. 7(a),Fig. 7(b), and Fig. 7(c). In detail, Fig. 7(a), colored in blue, depicts the process of the `pharmaceutical company` from the moment the drug order is received to its fulfillment. Figure 7(b), colored in green, depicts the process of the `Specialized clinic` from the moment the patient arrives from the `Hospital` to his transfer.

6.3 Memory Usage

7 Conclusion and Future Work

It can be reduced

In our implementation, we have focused on process discovery tasks. However, our approach has the potential to seamlessly cover a wider array of process mining functionalities such as *conformance checking*, and *performance analysis* techniques. Implementing them and show their integrability with our approach paves the path for future work.

To be rephrased and repositioned wherever it fits best.

Confidentiality is paramount in inter-organizational process mining, as sensitive data traverses organizational boundaries. Preserving the privacy and confidentiality of operational data becomes a critical concern in this regard. Our research explores a secrecy-preserving approach in which trusted applications allow organizations to apply process mining techniques using event logs from various organizations while ensuring the preservation of partners' privacy. Our solution still has room for improvement. Currently, we assume that providers act fairly, and we do not expect to have injected or maliciously manipulated event logs. In addition, we do not handle TEE crashes and suppose that miners and providers exchange messages in perfect communication channels where no loss, no snap, and no bit corruption take place. We also make assumptions on event log data. We assume the existence of a universal clock for event timestamps across various systems, eliminating the need for synchronization procedures. Additionally, we presume that traces from different organizations relating to the same process instance share a common case identifier. However, this assumption is unrealistic in real-world scenarios, where organizations might employ different case notations. To address this challenge, we should explore alternative event log representations. Future work includes the elaboration of an interaction protocol that formalizes the communication workflow between data providers and miners. Additionally, we plan to integrate usage control policies containing terms and conditions on event log utilization. To achieve this goal, we will design dedicated mechanisms inside trusted applications for monitoring usage rules and enforcing their fulfillment. The presented solution embraces model process mining techniques in a general way. However, we believe that the presented approach is particularly compatible with declarative model representations. Therefore, trusted applications could compute and store the entire set of rules representing a business process, and users may interact with them via trusted queries. We plan to extend the discussion in Sect. 6 by integrating threat modeling analysis and quantitative assessments concerning scalability, throughput, and performances on real-world event logs.

CDC: The bibliography entries are too rich. Look at [7]. Do we really care that the conference was in Toulouse? And look at [10]: the acronym is enough for the conference name. Also, the volume number is useless if we do not have the series (anyway, we could not care less about either of the two). We have already gone through this, so we should shorten the entries as we know.

References

1. van der Aalst, W.M.: Federated process mining: Exploiting event data across organizational boundaries. In: 2021 IEEE International Conference on Smart Data Services (SMDS). pp. 1–7 (2021). https://doi.org/10.1109/SMDS53860.2021.00011
2. Claes, J., Poels, G.: Merging event logs for process mining: A rule based merging method and rule suggestion algorithm. Expert Systems with Applications **41**(16), 7291–7306 (2014)
3. Cramer, R., Damgård, I.B., et al.: Secure multiparty computation. Cambridge University Press (2015)
4. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management, Second Edition. Springer (2018)
5. 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 2020, 25th International Conference, EMMSAD 2020, Held at CAiSE 2020, Grenoble, France, June 8–9, 2020, Proceedings 21. pp. 166–181. Springer (2020)
6. Elkoumy, G., Fahrenkrog-Petersen, S.A., Dumas, M., Laud, P., Pankova, A., Weidlich, M.: Shareprom: A tool for privacy-preserving inter-organizational process mining. BPM (PhD/Demos) **2673**, 72–76 (2020)
7. Engel, R., Krathu, W., Zapletal, M., Pichler, C., van der Aalst, W.M.P., Werthner, H.: Process mining for electronic data interchange. In: EC-Web. pp. 77–88 (2011)
8. Engel, R., Krathu, W., Zapletal, M., Pichler, C., Bose, R.J.C., van der Aalst, W.M.P., Werthner, H., Huemer, C.: Analyzing inter-organizational business processes: Process mining and business performance analysis using electronic data interchange messages. Inf. Syst. E-Bus. Manag. **14**, 577–612 (2016)
9. Hernandez-Resendiz, J.D., Tello-Leal, E., Marin-Castro, H.M., Ramirez-Alcocer, U.M., Mata-Torres, J.A.: Merging event logs for inter-organizational process mining. In: New Perspectives on Enterprise Decision-Making Applying Artificial Intelligence Techniques, pp. 3–26. Springer (2021)
10. Koch, N., Kraus, A.: The expressive power of uml-based web engineering. In: Second International Workshop on Web-oriented Software Technology (IWWOST02). vol. 16, pp. 40–41. Citeseer (2002)
11. Mannhardt, F.: Sepsis cases - event log (2016). https://doi.org/10.4121/UUID:915D2BFB-7E84-49AD-A286-DC35F063A460, https://data.4tu.nl/articles/_/12707639/1
12. Müller, M., Simonet-Boulogne, A., Sengupta, S., Beige, O.: Process mining in trusted execution environments: Towards hardware guarantees for trust-aware inter-organizational process analysis. In: International Conference on Process Mining. pp. 369–381. Springer International Publishing Cham (2021)
13. Zhao, C., Zhao, S., Zhao, M., Chen, Z., Gao, C.Z., Li, H., Tan, Y.a.: Secure multiparty computation: theory, practice and applications. Information Sciences **476**, 357–372 (2019)