CEUR-WS.org/Vol-3783/paper_324.pdf

CONFINE: Preserving Data Secrecy in Decentralized Process Mining

Valerio Goretti[1], Davide Basile[1], Luca Barbaro[1] and Claudio Di Ciccio[2]

[1] Sapienza Univeristy of Rome, Italy
[2] Utrecht University, Netherlands

Abstract

In the contemporary business landscape, collaboration across multiple organizations offers a multitude of opportunities, including reduced operational costs, enhanced performance, and accelerated technological advancement. The application of process mining techniques in an inter-organizational setting, exploiting the recorded process event data, enables the coordination of joint effort and allows for a deeper understanding of the business. Nevertheless, considerable concerns pertaining to data confidentiality emerge, as organizations frequently demonstrate a reluctance to expose sensitive data demanded for process mining, due to concerns related to privacy and security risks. The presence of conflicting interests among the parties involved can impede the practice of open data sharing. To address these challenges, we propose our approach and toolset named CONFINE, which we developed with the intent of enabling process mining on process event data from multiple providers while preserving the confidentiality and integrity of the original records. To ensure that the presented interaction protocol steps are secure and that the processed information is hidden from both involved and external actors, our approach is based on a decentralized architecture and consists of trusted applications running in Trusted Execution Environments (TEE). In this demo paper, we provide an overview of the core components and functionalities as well as the specific details of its application.

Keywords

Process mining, Decentralized computing, Confidential Computing, Trusted Execution Environment, Privacy

Metadata description	Value
Tool name	CONFINE
Current version	1.0
Legal code license	Apache 2.0
Languages, tools and services used	Go, Python
Supported operating environment	GNU/Linux
Download/Demo URL	github.com/Process-in-Chains/CONFINE.git
Documentation URL	github.com/Process-in-Chains/CONFINE/blob/main/README.md
Source code repository	github.com/Process-in-Chains/CONFINE
Screencast video	youtu.be/Oaoo6gS_4tw?si=hY0ztcbUrGO8PjIr

ICPM 2024 Tool Demonstration Track, October 14-18, 2024, Kongens Lyngby, Denmark

Corresponding author.

These authors contributed equally.

0000-0001-9714-4278 (V. Goretti); 0000-0002-5804-4036 (D. Basile); 0000-0002-2975-5330 (L. Barbaro); 0000-0001-5570-0475 (C. D. Ciccio)

© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

1. Introduction

Collaboration between multiple organizations is essential in today's highly competitive and development-oriented business environment. By aligning around shared goals, organizations can streamline operations, reduce redundancies, and ultimately improve efficiency, performance, and cost-effectiveness. In this context, inter-organizational process mining enables the coordination of joint efforts, improves overall transparency, and allows for benchmarking [1]. Nevertheless, despite the numerous benefits, a number of potential issues may arise, primarily related to data confidentiality. Information is an asset, and even in a collaborative environment companies are unwilling to share sensitive data required to execute process mining algorithms with their partners [2]. Allowing sensitive operational data to move across organizational boundaries inevitably raises issues related to data privacy and security, potentially exposing the data to unauthorized access and misuse [3]. For instance, we may consider a hospitalization process that involves three different parties: a hospital, a pharmaceutical company, and a specialized clinic. In this scenario, two other entities wish to uncover information on the inter-organizational process for reporting and auditing purposes: the National Institute of Statistics of the country where the three organizations reside and the University that hosts the hospital [4]. The hospital, specialized clinic, and pharmaceutical company have a partial view of the overall dynamics of the inter-organizational process as they record operations pertaining to their own parts. As a result, each player stories a separate event log partition. It would be mutually beneficial for all parties involved to have access to the findings of an aggregate data analysis, integrating the event log partitions yielded by the collaborating organizations. Nevertheless, an intrinsic divergence of interests emerges: granting access to traces to other organizations may reveal information the parties do not want to disclose. The ability to conduct process mining on data from multiple sources must, therefore, meet the necessity to safeguard the privacy of the entities involved and to guarantee the confidentiality of the information, ensuring that the local event log is never given away completely in-clear.

To solve this conundrum, we present CONFINE, our recently developed approach and toolkit designed to enhance collaborative information system architectures with secrecy-preserving process mining capabilities. To secure information secrecy during the exchange and elaboration of data, our solution resorts to *Trusted Execution Environments* (TEEs) [5]. These are hardware-secured contexts, which guarantee code integrity and data confidentiality before, during, and after their utilization. Owing to these characteristics, CONFINE lets information be securely transferred beyond an organization's perimeter. Therefore, computing nodes other than the information provisioners can aggregate and elaborate the original, unaltered process data in a secure, externally inaccessible vault. Also, CONFINE is capable of providing these guarantees while demanding low computation overhead and providing scalability.

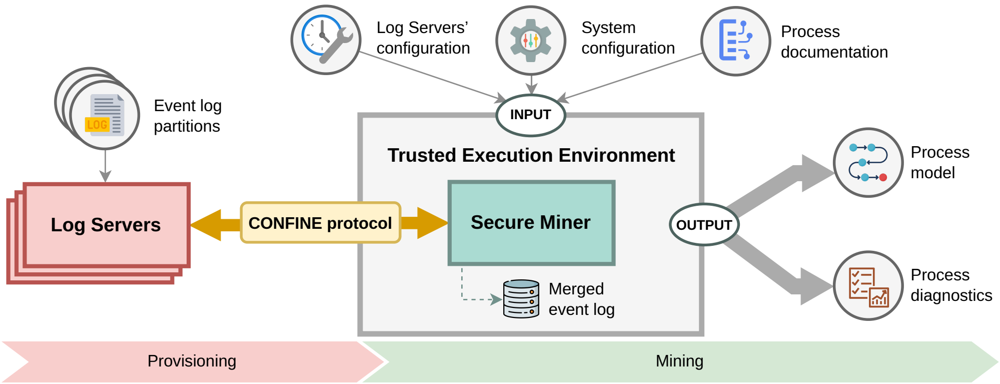


Figure 1: An overview of the CONFINE architecture

2. The CONFINE Framework

In the following, we present the core concepts of CONFINE by introducing the main components and discussing how they interact in the CONFINE protocol.

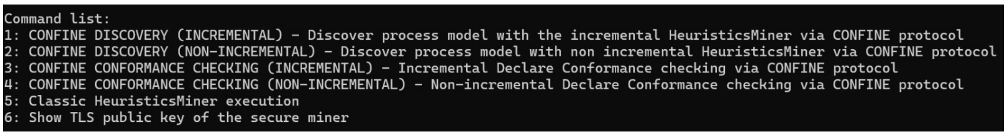
High-level architecture. Figure 1 displays a high-level schematization of the CONFINE framework. Our architecture involves different information systems running on multiple machines. An organization can take at least one of the following roles, depending on the tasks they take on: **provisioning** if it delivers local partitions of event logs to be collaboratively mined (e.g., the hospital in our example); **mining** if it applies process mining algorithms using event logs retrieved from provisioners (e.g., the National Institute of Statistics). In our solution, every organization hosts one or more nodes, incorporating components that depend on the roles played. The nodes hosted by provisioning organizations implement the `Log Server` component. The data provided by `Log Servers` is fed into the `Secure Miner` components, which are deployed on nodes hosted by mining organizations. Notice that every `Secure Miner` retrieves process data from one to many `Log Servers`, as each of the latter can offer different partitions of the event log (for example, the hospital records activities that are not visible to the pharmaceutical company, and vice-versa). In CONFINE, multiple organizations can perform mining tasks by hosting one or more `Secure Miners`, eliminating the need to depend on a central authority. This reflects the decentralized nature of our approach.

CONFINE allows miners to request data from provisioners that are part of the inter-organizational context to perform mining operations while ensuring the secrecy and confidentiality of the event logs exchanged. These properties are ensured since the `Secure Miner` is a trusted application executed within a trusted execution environment (TEE). TEEs create isolated contexts separate from the operating system, safeguarding code and data with hardware-based encryption mechanisms. They use dedicated CPU instructions to manage encrypted data within a reserved memory portion [6]. The CPU encrypts this memory with a random decryption key generated at each power cycle. By enforcing strict memory access controls, TEEs prevent applications from accessing or altering each other's memory space, thus enhancing system security. The CONFINE protocol securely interconnects `Log Servers` with the `Secure Miners`.

The CONFINE protocol. In our protocol, the interaction between components is divided into four sequential steps, namely *(i) initialization*, *(ii) remote attestation*, *(iii) data transmission*, and *(iv) computation.* The `Secure Miner` starts the protocol execution with the *initialization* step, during which it gets informed about the case distribution of the log partitions in the `Log Servers`; this data request includes identity evidence of the mining organization. In this phase, e.g., the `National Institute of Statistics` requests the hospital's `Log Server` to list aggregate information about the cases they can share in order to pre-plan the subsequent data requests. Next, the *remote attestation* phase occurs. The purpose of remote attestation is to establish trust between the `Log Servers` and the `Secure Miner`. Since the actual process data are going to be transmitted next, a stronger certification is required. This phase is thus based on the RATS RFC standard [7] (the basis of TEE attestation schemes) and has three main objectives: *(i)* to provide the `Log Servers` evidence that the data request for a log partition originates from a trusted application running within a TEE; *(ii)* to verify that the trusted application is indeed the authentic `Secure Miner` software entity; *(iii)* to identify the owner of the `Secure Miner` application. Once the trusted nature of the `Secure Miner` is verified, it retrieves from the `Log Servers` their log partitions to internally merge the cases. Based on the information retrieved during the initialization phase, it plans the requests accordingly. Each `Log Server` retrieves its event log and filters it based on the case id specified by the miner. Given the typically limited capacity of a TEE working memory, `Log Servers` are required to divide the filtered log into separate segments of up to a certain size. In the *computation* phase, the process mining technique selected by the organization is applied to the received data. The mining organizations can use the `Secure Miner` in either an *incremental* or a *non-incremental* manner. The distinction between these approaches concerns the timing in which the computation phase is executed. With the incremental approach, the `Secure Miner` begins the computation phase concurrently with the data transmission phase: As soon as all `Log Servers` have sent their log segments pertaining to a specific case, the mining algorithm is executed to produce and update a partial mining result. In contrast, with the non-incremental approach, the `Secure Miner` waits until all full partitions are received from the `Log Servers`, and only then it starts the mining algorithm. The incremental approach tends to save on TEE memory (the full log does not need to get loaded before mining begins), but it poses a restriction on the mining algorithm in use as it must be apt for treating partial inputs and produce updates.

3. Maturity

To showcase the capabilities of CONFINE, we propose a prototype implementation and run it with artificially generated and real-world data. Our `Secure Miner` implementation



```
Command list:
1: CONFINE DISCOVERY (INCREMENTAL) - Discover process model with the incremental HeuristicsMiner via CONFINE protocol
2: CONFINE DISCOVERY (NON-INCREMENTAL) - Discover process model with non incremental HeuristicsMiner via CONFINE protocol
3: CONFINE CONFORMANCE CHECKING (INCREMENTAL) - Incremental Declare Conformance checking via CONFINE protocol
4: CONFINE CONFORMANCE CHECKING (NON-INCREMENTAL) - Non-incremental Declare Conformance checking via CONFINE protocol
5: Classic HeuristicsMiner execution
6: Show TLS public key of the secure miner
```

Figure 2: Screnshot of the `Secure Miner`'s terminal interface

is an Intel-SGX[1] trusted app developed in GO.We adopt the EGo[2] framework to deploy the `Secure Miner` trusted app into the Intel SGX TEE. As depicted in Fig. 1, the `Secure Miner` takes as input a configuration of `Log Server`s, and a set of parameters to execute the protocol, and process documentation (a collective term indicating cues for the mining algorithms, such as a collaborative process specification for conformance checking). Users can interact with `Secure Miner` using a terminal interface through which commands are forwarded to the TEE. Figure 2 shows a screenshot of this interface. We developed the `Log Server` in GO. Communication between the `Secure Miner` and `Log Server` relies upon the HTTP protocol secured via TLS.

Our `Secure Miner` implementation incorporates the *Heuristics Miner* [8] algorithm for process discovery and a porting of the PM4Py *Conformance Declare*[3] algorithm for the conformance checking of declarative process specifications [9]. The results produced by the HeuristicsMiner can be examined using workflow net visualization tools like *WoPeD* [10]. We provide both an *incremental* variant, which updates partial results as complete cases are collected, and a *non-incremental* variant, starting only after the entire merged event log is collected. We validated our prototypical implementation of CONFINE on real-world event logs, including BPIC 2013 [11] and Sepsis [12], as well as a synthetic log generated from a healthcare scenario. We evaluated the convergence of the CONFINE protocol and the scalability of the `Secure Miner`'s runtime memory usage. These tests were conducted in both simulation mode and SGX native mode.

We plan to improve the CONFINE toolset in different directions. Given the stringency of the hardware requirement imposed by CONFINE, we envision real-world production scenarios involving the `Secure Miner` instances being deployed on remote machines owned by service providers. To take steps towards this direction, we plan to implement remote APIs through which mining organizations can remotely submit commands to the `Secure Miner`s and securely receive the output of the computation. At present, the `Secure Miner` prototype offers a limited choice of process mining algorithms. We aim to widen the spectrum of applicable algorithms by designing a plug-in integration system enabling implementers to add computation modules that are auditable by `Log Server` via *remote attestation*. Furthermore, the integration of a graphical user interface would facilitate the interaction with the `Log Server` and the `Secure Miner` trusted app. These improvements pave the path for future work.

[1] https://www.intel.com/content/www/us/en/developer/tools/software-guard-extensions/overview.html. Accessed: September 26, 2024.

[2] https://www.edgeless.systems/products/ego. Accessed: September 26, 2024.

[3] https://processintelligence.solutions/static/api/2.7.11/pm4py.algo.conformance.declare.html. Accessed: September 26, 2024.

4. Availability

A more detailed description of the CONFINE approach is available in [13]. Our implementation can be downloaded from github.com/Process-in-Chains/CONFINE. The `readme` file of the repository guides the installation of the `Secure Miner` and the `Log Server` components. The video demonstration of CONFINE is included in the repository's `readme` and can be directly watched at www.youtube.com/watch?v=Oaoo6gS_4tw.

Acknowledgments

This research work was partly funded by MUR under PRIN grant B87G22000450001 (PINPOINT), the Latium Region under PO FSE+ grant B83C22004050009 (PPMPP), Sapienza University of Rome under grant RG123188B3F7414A (ASGARD), and by the EU-NGEU under the NRRP MUR grant PE00000014 (SERICS).

References

[1] W. M. P. van der Aalst, Intra-and inter-organizational process mining: Discovering processes within and between organizations, in: PoEM, 2011, pp. 1–11.

[2] C. Liu, Q. Li, X. Zhao, Challenges and opportunities in collaborative business process management: Overview of recent advances and introduction to the special issue, Inf. Syst. Front. 11 (2009) 201–209.

[3] M. Müller, N. Ostern, Koljada, et al., Trust mining: analyzing trust in collaborative business processes, IEEE Access (2021) 65044–65065.

[4] M. Jans, M. Hosseinpour, How active learning and process mining can act as continuous auditing catalyst, Int. J. Accounting Inf. Systems 32 (2019) 44–58.

[5] M. Sabt, M. Achemlal, A. Bouabdallah, Trusted execution environment: What it is, and what it is not, in: 2015 IEEE TrustCom/BigDataSE/ISPA, 2015, pp. 57–64.

[6] V. Costan, S. Devadas, Intel SGX explained, Cryptology ePrint Archive (2016).

[7] H. Birkholz, D. Thaler, M. Richardson, et al., Remote ATtestation procedureS (RATS) Architecture, 2023.

[8] A. J. M. M. Weijters, W. M. P. van der Aalst, A. K. Alves De Medeiros, Process mining with the HeuristicsMiner algorithm, 2006.

[9] C. Di Ciccio, M. Montali, Declarative process specifications: Reasoning, discovery, monitoring, in: Process Mining Handbook, Springer, 2022, pp. 108–152.

[10] T. Freytag, P. Allgaier, A. Burattin, A. Danek-Bulius, Woped - A "proof-of-concept" platform for experimental BPM research projects, in: BPM (Demos), 2017.

[11] W. Steeman, BPI challenge 2013, incidents, 2013. doi:10.4121/UUID:500573E6-ACCC-4B0C-9576-AA5468B10CEE.

[12] F. Mannhardt, Sepsis cases - event log, 2016. doi:10.4121/UUID:915D2BFB-7E84-49AD-A286-DC35F063A460.

[13] V. Goretti, D. Basile, L. Barbaro, C. Di Ciccio, Trusted execution environment for decentralized process mining, in: CAiSE, Springer, 2024, pp. 509–527.