Shareprom: A Tool for Privacy-Preserving Inter-Organizational Process Mining

Gamal Elkoumy[1], Stephan A. Fahrenkrog-Petersen[2], Marlon Dumas[1], Peeter Laud[3], Alisa Pankova[3], and Matthias Weidlich[2]

[1] University of Tartu, Tartu, Estonia
{gamal.elkoumy,marlon.dumas}@ut.ee
[2] Humboldt-Universität zu Berlin, Berlin, Germany
{fahrenks,weidlima}@hu-berlin.de
[3] Cybernetica, Tartu, Estonia
{peeter.laud,alisa.pankova}@cyber.ee

Abstract. Process mining is a set of techniques to analyze business processes based on event logs extracted from information systems. Existing process mining techniques are designed for intra-organizational settings, as they assume that the entire event log of a process is available for analysis at once. In an intra-organizational process, each party only has access to its own private event log, which gives only a partial picture of the whole process. Moreover, the involved parties may be unwilling or unable to share their private logs due to confidentiality or privacy imperatives. In this context, this paper presents a tool, namely Shareprom, that enables independent parties to execute process mining operations without revealing any data other than the output of the analysis. Specifically, Shareprom uses secure multi-party computation techniques to compute the frequency or time-annotated Directly-Follows Graph (DFG) of event logs held by multiple parties, without these parties having to share any information other than the resulting DFG. The tool applies a differentially private release mechanism before revealing the output DFG in order to provide an additional layer of privacy protection.

1 Introduction

Process mining techniques enable organizations to analyze business processes from event logs extracted from information systems. These techniques allow us, for example, to identify performance bottlenecks and to recommend changes to a process in order to increase its efficiency. Existing process mining techniques assume that the event log to be analyzed can be accessed in its entirety. This assumption is reasonable in an intra-organizational setting, where all the data about a given process is located within the same trust domain. However, in an inter-organizational process, the event log is distributed across multiple independent organizations. For example, in a supply chain process, a manufacturer interacts with multiple suppliers. Each of them holds its own private event log, which only provides a partial view of the whole process.

Due to confidentiality concerns and privacy regulations, such as GDPR[4] and HIPAA[5], the involved organizations are unwilling to, or sometimes legally prevented from, sharing their event logs. They are prevented from sharing the logs either with each other or with a third-party, such as an independent analysis firm, to jointly perform the analysis.

In this paper, we present Shareprom, a secure multi-party computation system for inter-organizational process mining. Shareprom allows independent cooperating organizations to control what is disclosed from their event logs. We assume that the parties have agreed to reveal the time-annotated process map. In other words, the analysis firm will not learn anything about the independent organizations other than this output. Shareprom applies differential privacy to the resulting DFG so as to provide an extra layer of privacy, as presented in Section 2. The analysis firm will neither have access to specific traces, nor to any information associated to these traces. And naturally, the data providers will not be able to learn anything about each other's private data.

2 Overview of the Tool

Shareprom uses secure multi-party computation (MPC) to process event logs and to build the DFG matrix. Before revealing the DFG matrix, Shareprom applies Laplacian differential privacy by the injection of noise into the matrix. Shareprom relies on a platform for secret-sharing based MPC (specifically Sharemind [2]).

Shareprom uses the three-party MPC protocol set of Sharemind, which is secure against honest-but-curious adversaries. This means that as long as the parties are following the protocols honestly and do not collude, none of them will learn more than the size of the data. Parties in a typical secure MPC deployment can be: *input parties*, *computation parties*, or *output parties* [4]. The sets of input and computing parties may intersect. In this paper, we assume a simplified scenario, where the three computing parties themselves provide the inputs and receive the outputs. Two of the parties contribute the input data, and the third party is the analysis firm that receives the final output.

We assume that input parties share with each other the number of activities and the maximum trace length in their event logs. Also, we assume that the case identifiers are the same across the input parties. This is needed to conduct preprocessing that reduces the amount of information that might be learned from the size of data. Even with encrypted data, contextual knowledge might lead to leakage of some information [5]. An adversarial party might learn the shortest or the longest trace, and using the domain knowledge, they could reveal the actual activities. For such a case, we apply padding to the logs on the client side of each input party, so the resulting log has all the traces with the same length, which is set to the maximum trace length. Parties can hide the maximum trace length by adding extra padding. The logs are uploaded to computation servers in a secret-shared manner, and the MPC protocol performs computation without

[4] https://eur-lex.europa.eu/eli/reg/2016/679/oj
[5] https://www.hhs.gov/hipaa/

any intermediate declassifications. As such, during the computation, the parties do not learn anything in addition to the sizes of padded logs. This also excludes any attacks related to access patterns, like frequent pattern mining, which would be possible, if the events that are equal to each other, are leaked.

Figure 1 gives an overview of the system components. Below we summarize the functionality of each component of Shareprom. More information about the system components can be found in [3].

Preprocessing. Each party of the cooperating organization uses Shareprom clients to import the XES file of their event log. Shareprom then performs preprocessing of the event log at its organization site. The parties exchange the maximum trace length and the number of unique activities of their entire log. All the traces need to be padded to have the same length as the longest trace. The activities are mapped to a one-hot encoding format, independently on each input party. Also, a sort step of the logs by traces is performed. The two latter steps are needed as a preparation for the subsequent secure MPC protocol, which requires the data to be available in a specific format.

Mapping Event Log to Secret-shares. Each party uses their Shareprom client to map their event logs into secret-shares. Each client pushes its secret-shares to the Sharemind servers. Secret-shares do not reveal any information about the event log. Until this point, the analysis firm, as one of the computation parties, has only received a single share of each input, which alone looks like a random value.

DFG Matrix Calculation. Shareprom runs the Sharemind MPC protocol to construct the DFG matrix from uploaded secret-shared data. An algorithm based on a one-hot encoding technique allows us to ensure that the computation does not reveal any information to the parties, including the access pattern (e.g. which log entries follow each other). The details of this algorithm can be found in [3]. The resulting DFG takes the form of a matrix with secret-shared entries, where each entry corresponds to a transition between two activities, containing the frequency count and the total execution time for that transition.

Differential Privacy. The DFG itself may reveal some information about the parties' inputs. We consider the frequency and time difference between each pair of activities in the DFG as a separate query. Hence, before disclosing the final result to the analyst party, we enhance it with differential privacy by injecting Laplacian noise to the frequency and time difference of each pair of activities. The added noise conceals the order of activities and their execution times. We consider understanding the trade-offs between the amount of added noise and the utility of the output as a direction for future work.

3 Tool Packaging and Maturity

Shareprom is developed in Python 3.7 and SecreC – a dialect of the C language supported by the Sharemind plartform [1]. For ease of use, Shareprom comes with

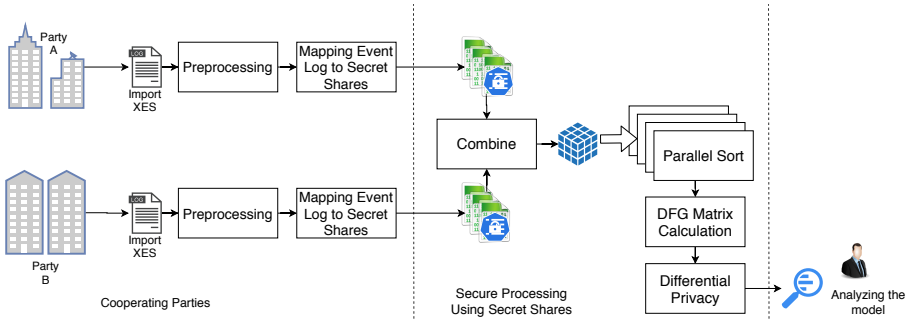


Fig. 1: Overview of Shareprom

a desktop client application. The desktop client allows a user to import an event log (in XES format) as shown in Figure 2a. The desktop client connects with its respective Shareprom server. In a typical configuration, three Sharemind servers are inter-connected, e.g. two input providers and a computation node – the latter representing for example an analysis firm, as shown in Figure 2b. The analysis firm can view the output (with differential privacy noise) as shown in Figure 2c. An analyst at this firm can then analyze the output and share the findings (or the whole output) with the input providers depending on the intended use case.

Shareprom has been tested with event logs of different characteristics, as reported in [3]. The tool can handle event logs with several thousands of events with up to a couple of dozens distinct activity labels. Further optimizations targeted at specific use cases should allow it to scale to large logs in future.

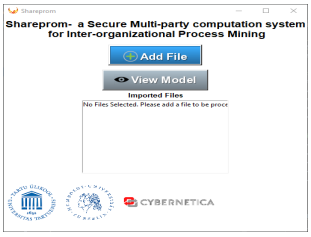
4 Screencast and Source Code

A screencast is available at `https://youtu.be/uz2mrYz-y-w` . This screencast illustrates a scenario where two parties (a manufacturer and a supplier) delegate an analysis task to a data analysis firm. The manufacturer produces goods based on purchase orders. Parts of the production pipeline need intermediate parts from the supplier. The manufacturer orders the intermediate parts from the supplier. The supplier produces the intermediate parts and transports them to the manufacturer. Each party has its own separate event log. The analysis firm analyzes the joint DFG constructed from the event logs.

The source code, installation steps, dependencies, and example event logs can be found at `https://github.com/Elkoumy/shareprom/releases/tag/v0.2`.

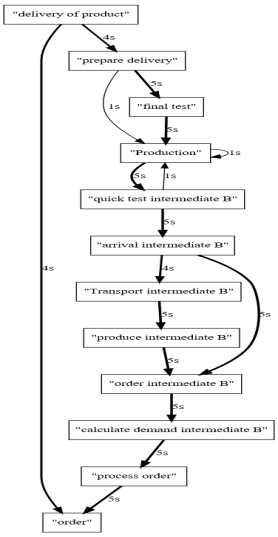
5 Conclusion

Shareprom leverages the secure multi-party computation capabilities of Sharemind in order to enable multiple parties to perform inter-organizational process mining without revealing any data other than the analysis output. A differential privacy



(a) Shareprom interface for Organizations

(b) Shareprom interface for Analysis Firm



(c) The Joint DFG After Differential Privacy

Fig. 2: Shareprom Interface.

mechanism allows this output to be further protected against possible privacy leakages. At present, Shareprom focuses on the computation of DFGs, which is a basic graphical process representation used in process mining tools. Future work aims at expanding the capabilities of Shareprom to other process mining operations, while at the same time tackling associated scalability challenges.

Acknowledgments. This research is partly funded by ERDF via the Estonian Centre of Excellence in ICT (EXCITE).

References

1. Bogdanov, D., Laud, P., Randmets, J.: Domain-polymorphic programming of privacy-preserving applications. In: Proceedings of the Ninth Workshop on Programming Languages and Analysis for Security. pp. 53–65 (2014)
2. Bogdanov, D., Laur, S., Willemson, J.: Sharemind: A framework for fast privacy-preserving computations. In: European Symposium on Research in Computer Security. pp. 192–206. Springer (2008)
3. Elkoumy, G., Fahrenkrog-Petersen, S.A., Dumas, M., Laud, P., Pankova, A., Weidlich, M.: Secure multi-party computation for inter-organizational process mining. In: Proceedings of the BPMDS and EMMSAD Working Conferences. Springer (2020)
4. Kamm, L.: Privacy-preserving statistical analysis using secure multi-party computation. Ph.D. thesis, University of Tartu (2015)
5. Rafiei, M., von Waldthausen, L., van der Aalst, W.M.: Supporting confidentiality in process mining using abstraction and encryption. In: Data-Driven Process Discovery and Analysis, pp. 101–123. Springer (2018)