Preserving Confidentiality in Data Analytics-as-a-Service

Tillem, Gamze

DOI
10.4233/uuid:2332e125-c9c4-443c-9a18-0f8fd1c2f85e

Publication date
2020

Document Version
Final published version

Citation (APA)
Tillem, G. (2020). *Preserving Confidentiality in Data Analytics-as-a-Service*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:2332e125-c9c4-443c-9a18-0f8fd1c2f85e

Preserving Confidentiality in Data Analytics-as-a-Service

Gamze Tillem

PRESERVING CONFIDENTIALITY IN DATA ANALYTICS-AS-A-SERVICE

PRESERVING CONFIDENTIALITY IN DATA ANALYTICS-AS-A-SERVICE

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus Prof. dr. ir. T.H.J.J. van der Hagen,
chair of the Board for Doctorates
to be defended publicly on
Wednesday 20 May 2020 at 10:00 o'clock

by

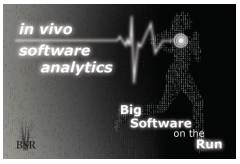
Gamze TİLLEM

Master of Science in Computer Science and Engineering,
Sabancı University, Istanbul, Turkey,
born in Denizli, Turkey.

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Prof. dr. ir. R.L. Lagendijk,	Delft University of Technology, promotor
Dr. Z. Erkin,	Delft University of Technology, copromotor
Independent members:	
Prof. dr. A. van Deursen	Delft University of Technology
Prof. dr. M.J. van den Hoven	Delft University of Technology
Prof. dr. ir. G.J.P.M. Houben	Delft University of Technology
Prof. dr. M. Conti	University of Padua, Italy
Assoc. prof. dr. M. Önen,	Eurecom, France



Keywords: data analytics, secure computation, confidentiality

Printed by: IPSKAMP Printing

Cover design: Merve Tillem

ISBN 978-94-028-2044-7

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

CONTENTS

SUMMARY

The enhancements in computation technologies in the last decades enabled businesses to analyze the data that is collected through their systems which helps to improve their services. However, performing data analytics remains a challenging task for small- and medium-scale companies due to the lack of in-house experience and computational resources. Data Analytics-as-a-Service (DAaaS) paradigm provides such companies outsourced data analytics, where a company that is specialized in data analytics serves its knowledge and computational resources to the other companies, which need data analytics for their businesses.

A major challenge in DAaaS is preserving the privacy of the outsourced data, which might contain sensitive customer or employee information or the intellectual property of the outsourcing company. Leakage of sensitive information has several consequences both for outsourcing and service provider companies as legal obligations, loss of reputation, and financial loss. Therefore, a well functioning outsourced analytics service should achieve several data protection measures such as confidentiality, integrity, and availability.

In this thesis, we focus on the preservation of confidentiality in data analytics-as-a-service applications. We select three analytics applications that are becoming popular in outsourced data analytics, which are process analytics, machine learning, and marketing analytics. Despite there exist several other techniques that are commonly used in outsourced data analytics, we decide to focus on the algorithms of process analytics, machine learning, and marketing analytics since the privacy concerns in these analytics have not been investigated thoroughly.

In confidential data analytics-as-a-service, our goal is to achieve confidentiality by protecting input/output privacy and maintaining the correctness and efficiency of analytics computations. To protect the privacy of data we use two secure computation techniques, which are homomorphic encryption and secure multiparty computation. To assure correctness, we propose several hybrid protocol designs that minimize the loss of accuracy in computations. For the efficiency of our protocols, we use several optimization techniques that reduce the computation and communication costs of private data analytics. Our protocols show promising results for confidential data analytics in the outsourced setting.

SAMENVATTING

In de afgelopen decennia zijn de computatietechnologieën verbeterd. Dankzij deze verbetering hebben bedrijven gegevens kunnen analyseren die via hun systemen zijn verzameld, wat helpt om hun diensten te verbeteren. Echter, het uitvoeren van gegevensanalyse blijft een uitdagende taak voor het midden- en kleinbedrijf (MKB) te wijten aan het gebrek aan ervaringin en computationele middelen. Het Data Analytics-as-a-Service (DAaaS) paradigma biedt dergelijke bedrijven uitbestede gegevensanalyse, waarbij een bedrijf dat gespecialiseerd is in gegevensanalyse zijn kennis en computationele middelen levert aan de andere bedrijven, die gegevensanalyse nodig hebben.

Een grote uitdaging bij DAaaS is het beschermen van de privacy van de uitbestede gegevens, die gevoelige klant- of werknemersinformatie, of het intellectuele eigendom van het uitbestedende bedrijf kunnen bevatten. Het lekken van gevoelige informatie heeft verschillende gevolgen voor zowel uitbesteding als dienstverlenende bedrijven, zoals wettelijke verplichtingen, reputatieschade, en financieel verlies. Daarom moet een goed functionerende en uitbestede analyseservice verschillende gegevensbeschermingsmaatregelen treffen die leiden tot een gewenst niveau van vertrouwelijkheid, integriteit en beschikbaarheid.

In dit proefschrift richten we ons op het behoud van vertrouwelijkheid in DAaaS applicaties. We selecteren drie analysetoepassingen die populair zijn in uitbestede gegevensanalyse, namelijk procesanalyse, machine learning, en marketinganalyse. Ondanks dat er verschillende andere technieken bestaan die vaak worden gebruikt bij uitbestede gegevensanalyse, concentreren we ons op de algoritmen van procesanalyse, machine learning, en marketinganalyse, aangezien de privacykwesties in deze analyses niet grondig zijn onderzocht.

Bij vertrouwelijke DAaaS is ons doel om vertrouwelijkheid te bereiken door de privacy van input en output te beschermen en de juistheid en efficiëntie van analytische berekeningen te behouden. Om de privacy van gegevens te beschermen gebruiken we twee veilige berekeningstechnieken, namelijk homomorfe versleuteling en veilige berekening met meerdere partijen. Om de juistheid te verzekeren, stellen we verschillende hybride protocolontwerpen voor die het verlies aan nauwkeurigheid in berekeningen minimaliseren. Voor de efficiëntie van onze protocollen gebruiken we verschillende optimalisatietechnieken die de berekenings- en communicatiekosten voor gegevensanalyse. Onze protocollen laten veelbelovende resultaten zien voor vertrouwelijke gegevensanalyse in de uitbestede setting.

1

INTRODUCTION

Big data became a prominent term for businesses in the last decade with the dramatic increase in the amount of data generated which is enabled by innovations in computation technologies. In 2016, people generated 2.5 quintillion bytes of data every day [1]. By 2020, this number is expected to reach 146880 GB per person [2]. The massive amount of data generated by the computing systems does not remain idle. Companies collect and analyze the data to improve their services and products, for example, to understand customer behaviour, and reduce the risk of cybersecurity threats against their business [3]. Rather than seeing it as an auxiliary tool, companies have embraced data analytics as a booster for their businesses. As of 2018, the percentage of enterprises that adopted data analytics has reached to 59%, which goes over 90% in some industries, such as telecommunication [4].

Performing data analytics is not an easy task for companies. One challenge in utilizing data analytics is the lack of in-house experience. Especially small- and medium-size companies do not have employees who are specialized in data analytics. Companies can either recruit new employees, which incurs an additional financial cost, or train existing employees, which requires investment in time. The additional time and money investment make managers reluctant to fund data analysis teams [5]. Another challenge in in-house data analytics is the lack of computational resources. Without having the necessary computational infrastructure, investing in human resources is not adequate to perform successful data analytics. Furthermore, even if the enterprises have adequate human and computational resources, it is still very challenging for them to follow the latest advances in data analytics by themselves [6].

1.1. THE RISE OF DATA ANALYTICS-AS-A-SERVICE

Outsourcing data analytics tasks to an external company is a viable solution to overcome the inability of in-house data analytics. Inspired by the cloud computing services (Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS)), Data Analytics-as-a-Service (DAaaS) paradigm offers enterprises data

analytics services in an outsourced manner [7]. In this paradigm, a company which is specialized in data analytics serves its knowledge and computational resources to the companies that need data analytics for their businesses. The decrease in financial cost and the time spent makes data outsourcing attractive for companies. Delegating the analytics to an expert company results in higher quality analytics since the data analytics company owns the newest analytics tools and is aware of the innovations in the specific domain [6]. Furthermore, it strengthens the public accountability of companies since an external team can provide the correct results without any bias [6]. With all the advantages, outsourced analytics is widely adopted by companies such that by 2019, almost half of the data analytics tasks worldwide are outsourced [8].

It is important to note that data science specialists do not advise companies to outsource their data analytics tasks fully. The best practices in DAaaS suggest outsourcing certain type of analytics applications and build an in-house data analytics team for the rest of applications [6]. In this way, they can take faster action in emergency cases and still keep track of the newest technologies in data analytics. Below we discuss several outsourced analytics applications purchased by companies.

PROCESS ANALYTICS

Process analytics provides organizations insights in the effectiveness of their business processes [9]. By visualizing business processes companies can detect the bottlenecks in their systems and take corresponding actions. Performing process analytics is useful to analyze the workflow within the company, to clarify job functions for employees, and to conduct internal auditing tasks. Assuring the efficiency of processes is a high priority analytics task for many companies [10]. Process mining is a prominent mechanism to perform process analytics which aims to discover, monitor, and improve the real-life processes by extracting knowledge from the event data generated by digital systems [11]. Process mining is used for process analytics in three ways that are 1. *process discovery* which discovers a process model from raw event logs, 2. *conformance checking* which compares a process model with an event log to observe if the real behaviour in the log matches the expected behaviour in the process model, and 3. *process enhancement* which extends a process model with additional information and perspectives [11].

Conducting process analytics is not an easy task for many companies since the underlying techniques require them to have specific knowledge on process analytics which should be updated with the changing business dynamics and technologies [10]. Therefore, companies prefer to outsource the process analytics tasks to service providers that are the experts in the field. Outsourcing process mining tasks is becoming popular for many enterprises such that the market size is expected to easily triple or quadruple in the upcoming few years [12].

MACHINE LEARNING

The advances in computation technologies enabled machine learning techniques, especially neural networks, to reduce the long computation times to a feasible range which results in success in several fields such as image classification [13], voice recognition [14], and problem-solving [15]. Nowadays, machine learning techniques are used by enterprises to automate business functions such as root-cause analysis, targeted advertising, customer services, and forecasting [16].

Despite the success of machine learning that promises significant improvement in business functions, deploying machine learning is a big challenge for companies. Apart from having the knowledge to implement the techniques, acquiring computational resources and sufficient amount of data challenge companies to successfully use machine learning in their businesses. The solution to overcome the difficulties in the deployment of machine learning is possible by outsourcing the tasks to the service providers. The well-known technology companies such as Google [17], Amazon [18], IBM [19] lead the outsourcing market by providing their resources to the enterprises. The outsourcing market for machine learning services is promising such that it is expected to grow over 43% between 2019 and 2024 [20].

MARKETING ANALYTICS

Marketing is an important business function for companies to decide the best strategy to make money from their products. Like many other business functions, the digital era required companies to change their strategies for the marketing of their products. Apart from using the classical means of marketing such as television advertisements, billboards, or magazines, companies are now adopting online marketing practices to provide targeted advertisements for their customers which are determined by the previously observed behavior of the customer [21]. Using online personalized advertisements benefits companies by offering their services faster, easier and on a global range. The use of data analytics is important in digital marketing to understand customer behaviour, market trends, to decide pricing and also to get to know the competitors. Companies use several machine learning techniques such as logistic regression [22], deep neural networks [23], factorization machines [24] to match the right advertisement to the right customers by observing customer behaviour.

For most companies, surviving in the competitive advertising environment is possible by investing significant amount of money on marketing analytics. Between 2017 and 2018, the share of marketing analytics in marketing budget was 9.2% which constitutes the largest share of the budget [25]. Considering the drawbacks of performing in-house data analytics, companies prefer to outsourced their marketing analytics. By outsourcing marketing analytics, companies can save money with an improved quality of analytics [21]. Furthermore, outsourcing provides companies a fair competition area with their competitors since the analytics are performed by professional service providers.

1.2. PRIVACY CONCERNS IN DATA ANALYTICS-AS-A-SERVICE

One big challenge in data analytics-as-a-service is assuring privacy protection of outsourced data. The privacy-protection is important to protect the outsourced data, which can belong to the customers or employees of the company, the computation tasks, and the intellectual property of the company. Outsourcing companies have mainly three concerns when they are using DAaaS services [26]. The first one is trust in the service provider company. The companies believe that they lose control over their data when they share the data with external parties who are not necessarily trusted [26]. Since the company, which outsources the data analytics, is responsible to its customers and employees on the protection of sensitive data, a possible untrusted act by the service provider implies the charges on the outsourcing company, as well. A second concern is

leaking information to their competitors by sharing the same DAaaS service with multiple other customers. The service providers open their resources for multiple companies and provide them analytics simultaneously. Although the service providers are well aware of the regulations about the protection of each outsourcing companies' data and computation tasks, a possible breach can leak intellectual property to the competitors [26]. The third concern of outsourcing companies is regarding the technical mechanisms used to protect the data and analytics. With the growth in data sizes and the change in the complexity of analytics, the outsourcing companies want to assure the right mechanisms are used by the service providers to protect their data [26].

Leaking customer data or intellectual property has several consequences for analytics outsourcing companies and service providers. The most crucial consequence is the legal enforcements. Handling sensitive data requires conformance to several legal mechanisms; and a failure in data protection can cause sanctions and penalties imposed by the governments or the unions. The legal enforcements lead to the loss of reputation and public accountability for companies, which is followed by the financial consequences that lead to the loss of business and company value. An example of such privacy incident is encountered by Facebook after the Cambridge Analytica scandal due to deceiving the users about the control of their personal information [27]. The company is enforced to modify the corporate structure, submit new privacy restrictions, and pay $5 billion penalty, which is the largest ever imposed on any company for violating customer privacy [27]. After the announcement of the scandal, Facebook lost its market value by more than $36 billion in a couple of days [28]. Furthermore, 3 million European users deleted their accounts within a couple of months after the scandal leaked [29].

With all the lessons taken from the previous privacy incidents, as of May 2018, the companies are required to pay even more attention to the protection of data with the new General Data Protection Regulation (GDPR) in the EU [30]. Although there had been several effective data protection directives, the European Union designed the GDPR as a broader and deeper data protection regulation to protect individual's privacy rights in the digital era [31]. The regulation brings some changes in the extent of the application of protection laws, consent, penalties, and privacy-by-design. Accordingly, any company (inside or outside the EU) that works with the data related to the EU residents should comply with the regulation [32]. It requires the companies to give the conditions of consent in a clear and plain language [32]. The regulation handles the security breaches more strictly by the requirement of notification within 72 hours and with fines of up to €20 million or 4% of the annual global turnover [32]. Another important change introduced by the GDPR is the requirement of privacy-by-design which requires the companies to implement technical mechanisms for the protection of data apart from the legal contracts [32].

The changes required by the GDPR is valid for any type of activities that involves personal information. Therefore, it is important to look at what the GDPR implies for data and analytics outsourcing. The regulation requires the companies that outsource their analytics tasks to be more active in the control of the outsourced data and computations by imposing several technical and organizational procedures on the service provider company to protect the sensitive data [33, 34]. The service provider companies, on the other hand, should take measurements on the assurance of privacy-by-design with the

adoption of data protection techniques such as encryption or pseudonymization [35]. Selection of the appropriate data protection measure (encryption or psedonymization) is important to guarantee flawless data protection. Confidentiality, integrity, and availability, a.k.a. CIA triad of information security, are three major elements that guides the organizations in determining the technical data protection measures [36]. In Figure 1.1, we explain each element of the CIA triad.

CIA Triad

Confidentiality

Confidentiality assures that access to data is provided only to authorized parties. Outsourcing data and analytics to a service provider company weakens the control over the data for the outsourcing company. Thus, assuring confidentiality in data analytics-as-a-service requires hiding the outsourcing company's data and computation results from the service provider company and other customers [26].

Integrity

Integrity assures only authorized parties can modify the data. In outsourced analytics, both computation integrity and data integrity should be assured. Data integrity requires to preserve the trustworthiness of data without modifying or compromising, while computation integrity requires execution of computations without any interruptions that may or may not lead in any incorrect output [26].

Availability

Availability assures that any authorized party can access the data. Availability is crucial for outsourced data analytics since providing on-demand service is the core functionality of outsourcing [26]. A failure to access data or analytics discourages the outsourcing company to use outsourced analytics services.

Figure 1.1: Explanation of the elements of the CIA triad.

In a well functioning outsourced analytics service, achieving all elements of the CIA triad, i.e., confidentiality, integrity, and availability, is equally important. However, considering the privacy concerns of outsourcing companies and the requirements of the GDPR on data protection, guaranteeing confidentiality in DAaaS becomes a prominent

challenge for both outsourcing and service provider companies. Therefore, in this thesis, we focus on the design of protocols which target to achieve confidentiality in DAaaS. In the following section, we provide more information about confidentiality in outsourced data analytics.

1.3. CONFIDENTIAL DATA ANALYTICS-AS-A-SERVICE

In confidential data analytics-as-a-service, the goal is to enable outsourcing of data analytics tasks while assuring the confidentiality of outsourced data and the computation results. A typical confidential DAaaS setting consists of two parties which are an *analytics outsourcing company* and a *service provider company* as illustrated in Figure 1.2.

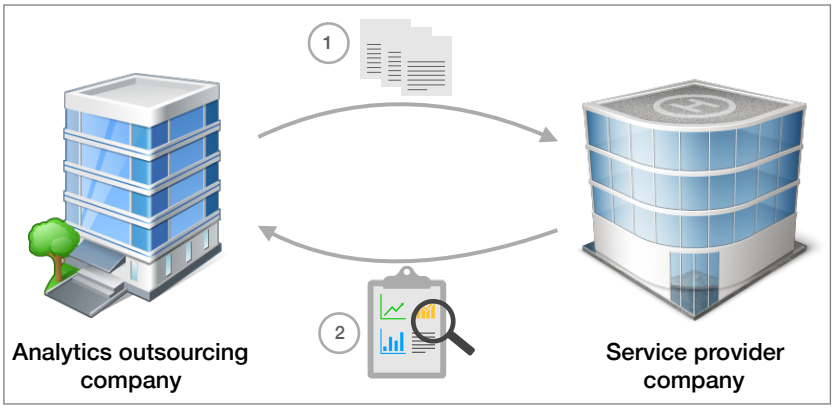


Figure 1.2: Parties involved in confidential data analytics-as-a-service.

- **Analytics outsourcing company** is a company that lacks the necessary knowledge, human power, or computational resources to apply data analytics. Therefore, the company outsources its analytics tasks to a service provider company. The analytics outsourcing company desires to protect the sensitive content of the outsourced data and computation results while benefitting from analytics services.
- **Service provider company** is a company that is specialized in performing several data analytics techniques and owns computational resources to perform the analytics. Its goal is to maximize its profit by offering its knowledge and computational resources to the outsourcing company by performing analytics for the company. As important as the business intelligence of the outsourcing company, in the computations of data analytics, the business intelligence of the service provider can also be sensitive, so, the service provider company might want to keep their algorithms confidential.

Designing a confidential data analytics protocol in the outsourced setting requires to assure several specifications, which protect the protocol against possible adversarial attempts. The first specification is to guarantee input and output privacy [37]. The service provider company should not be able to retrieve any information observing the

protected input provided by the analytics outsourcing company. Furthermore, after and during the execution of the protocol, the service provider company should not be able to obtain information about the output and the intermediary messages. If the service provider prefers to keep the company's business intelligence confidential, then the analytics outsourcing company also should not be able to retrieve any information about the business intelligence of the service provider.

The second specification is the correctness of the output, which necessitates verifying that the returned output is correct [37]. Verifying correctness is important when a corrupted service provider tries to alter the computation steps and the output. However if the adversary does not interrupt the execution, verifying correctness is not necessary. In some cases, even though the service provider does not maliciously interrupt the execution, the protocol may not return the expected output due to limitations on the flexibility of some operations, which is caused by the chosen data protection technique [38]. In such cases, rather than assuring correctness, achieving the highest accuracy becomes the goal of the service provider.

The third specification is providing efficiency in computation and communication costs. If the analytics outsourcing company is included in computations, the company's tasks should be minimized since its computational power is limited compared to the service provider. On the other hand, the cost of computations on the service provider should also be feasible such that the implemented privacy protection mechanisms do not add a significant overhead on computations [38].

The confidentiality specifications in DAaaS can be meet using cryptographic methods. Traditional encryption mechanisms, such as AES [39], are not sufficient for confidential DAaaS since they do not enable processing on protected data. However, there exist several modern cryptographic techniques that allow performing certain functionalities on protected data. We can the cryptographic techniques that are used in confidential DAaaS into three groups with respect to their functionalities as follows [38]:

- **Secure search on protected data:** Performing queries on protected data to retrieve a set of records is one major functionality of outsourced computation and storage. Searchable encryption [40] and order-preserving encryption [41] techniques provide efficient solutions to perform search on encrypted datasets with a trade-off of certain level of information leakage. Private information retrieval [42] and oblivious RAM [43] overcome the problem of information leakage in the former solutions. However, these solutions are usually not practical due to their significant computation and communication cost.

- **Secure computation on protected data:** Most data analytics operations require more complex functions than searching on protected data. Homomorphic encryption [44, 45] and secure multiparty computation [46, 47] are two techniques which enable to perform computations on secured data. Using homomorphic encryption, arithmetic operations can be computed on encrypted data without decryption. However homomorphic cryptosystems usually expensive with respect to the cost of computations. Secure multiparty computation, on the other hand, provides more flexibility in computations by allowing linear and nonlinear operations. The drawback of secure multiparty computation is its interactive nature

which might result in high bandwidth usage in computations.

- **Access control on protected data:** Restricting access to data or computation results is another important functionality of confidential DAaaS. Using public key encryption [48] is one method to grant access to someone who possesses the corresponding secret key for the public key. Identity-based encryption [49] and attribute-based encryption [50] are also used for access control which enables access to certain identities or attributes, respectively. Finally, functional encryption [51] grants access to the result of a function on the encrypted data.

1.4. PROBLEM STATEMENT

In this thesis, we aim to preserve confidentiality in data analytics-as-a-service applications. We choose three analytics applications that are becoming popular in outsourced data analytics, which are process analytics, machine learning, and marketing analytics. Despite there exist several other techniques that are commonly used in outsourced data analytics, we decide to focus on algorithms of process analytics, machine learning, and marketing analytics since the privacy concerns in these analytics have not been deeply investigated. For instance, in the field of process mining, no work achieves the confidentiality requirements and provides an efficient solution for process mining tasks. Similarly, the existing works in the online behavioral advertisement environment can only achieve partial privacy preservation with a focus on anonymity protection but the confidentiality of end-user data has not been achieved comprehensively. The research on private neural network operations proposes a handful amount of work on the protection of data and analytics results. However, the problems related to the accuracy, performance, and practicality requires deeper investigation of the research on private neural networks. In Chapter 2, we provide a detailed explanation related to these analytics applications and also review the existing literature that focuses on the protection of privacy in these analytics.

Considering the shortcomings of the literature on confidential DAaaS protocols, in this thesis, we aim to answer the following research question:

> *Which cryptographic techniques and optimization methods can be used to improve the computation and communication performance in confidential Data Analytics-as-a-Service while maximizing the accuracy of algorithms?*

We detail our research question with several subquestions that focus on the analytics applications we choose. Our subquestions are:

- *How efficiently can a service provider company perform process analytics in confidential DAaaS, where the accuracy of process analytics algorithms are maintained?*
- *How can the cost of computation and communication be balanced by a service provider company who performs private neural network operations in confidential DAaaS?*
- *What is the feasibility of operating Real-Time Bidding mechanism for online behavioral advertising using cryptographic techniques?*

1.5. CONTRIBUTION OF THE THESIS

The protocols we propose in this thesis offer privacy-preserving solutions for several common data analytics-as-a-service applications. In all of our protocols, we use secure computation techniques to protect and to process sensitive information of analytics outsourcing companies. The contributions of the thesis are as follows:

- To the best of our knowledge, we propose the first privacy-preserving protocols for process mining and for online behavioral advertisement.
  - In **Chapter 3**, we propose the first provably secure protocols for the discovery of business processes which proposes a comprehensive solution.
  - In **Chapter 4**, we propose the first protocol that executes conformance checking under privacy preservation.
  - In **Chapter 6**, we propose the first protocols that preserve privacy in online behavioral advertising which allow the usage of detailed user profiles and machine learning techniques.
- Our proposals are efficient with respect to computation and communication cost. To improve the performance of our protocols, we utilize several techniques such as data packing, single instruction multiple data operations, or multi-exponentiations. To the best of our knowledge, our proposal in **Chapter 5** is the first protocol which optimizes the nonlinear layers of private neural networks.
- We propose to use hybrid approaches which brings together different cryptographic techniques or different variants of the same cryptographic techniques. Using a hybrid mechanism, our protocols achieve higher accuracy since we are able to perform more flexible operations. Furthermore, we improve the efficiency of our protocols with respect to computation cost.
- We achieve the three requirements of confidentiality in analytics outsourcing in all of our protocols. The solutions we propose guarantees
  - **input and output privacy** by using provably secure cryptographic techniques for data protection and processing,
  - **accuracy** by successfully transforming analytics functions to protected domain or by combining different cryptographic mechanisms to increase flexibility of functions,
  - **efficiency** by minimizing the number of costly operations, using several optimization mechanisms and, when possible, utilizing a hybrid approach in protocols design.

1.5.1. OUTLINE

The structure of the thesis is as follows:

CHAPTER 2
CONFIDENTIAL DATA ANALYTICS-AS-A-SERVICE

Achieving privacy preservation in the outsourced data analytics is possible with a clear knowledge of the type of data analytics used and the cryptographic technique used. Therefore, in **Chapter 2**, first, we provide a preliminary explanation of the available cryptographic techniques that can be used in secure computation. Then, we introduce the most common algorithms used in the outsourcing of process analytics, marketing analytics, and machine learning. Furthermore, if available, we present the existing privacy-preserving solutions for the given analytics types and discuss their advantages and disadvantages. We conclude the chapter with a summary of open challenges in the existing solutions that are going to be addressed in the proceeding chapters.

CHAPTER 3
PROCESS DISCOVERY ON ENCRYPTED DATA

In process analytics, an important analytics task is to observe the processes within a company with respect to the activities performed, employees involved, and resources used in each process. Process mining offers several algorithms for discovering processes from logged data. However, the existing algorithms for the discovery of processes require privacy protection since the data might contain sensitive information of employees and customers. In **Chapter 3**, we present two protocols which assure privacy preservation in the discovery of processes. Our first protocol transforms a well-known process discovery algorithm, Alpha algorithm [52], to a privacy-preserving variant using homomorphic encryption. Our second protocol extends the first protocol by presenting a generalized approach that can be used as a basis for all existing process discovery algorithms. This chapter is an integral copy of the papers "*Mining Encrypted Software Logs using Alpha Algorithm*" by G. Tillem, Z. Erkin, and R.L. Lagendijk in *SECRYPT*. (pp. 267-274) (2017) and "*Mining Sequential Patterns from Outsourced Data via Encryption Switching*" by G. Tillem, Z. Erkin, and R.L. Lagendijk in *PST*. (pp. 1-10) (2018).

CHAPTER 4
PRIVACY-PRESERVING CONFORMANCE CHECKING FOR INTERNAL AUDITING

Another important task in process analytics is checking the compliance of the real behaviour of a system to the expected behaviour to detect deficiencies. Conformance checking is one technique, which checks whether the monitored behavior recorded in an event log complies with the normative behavior represented as a process model. In **Chapter 4**, we propose two protocols for privacy-preserving conformance checking which enables the companies to outsource their internal audit analytics to specialized service providers without leaking sensitive data of their employees. We use secure two-party computation that achieves promising performance results. This chapter is an integral copy of the paper "*Privacy-Preserving Conformance Checking for Internal Auditing*" by G. Tillem, N. Zannone, and Z. Erkin which is in preparation.

CHAPTER 5
PRIVATE NEURAL NETWORK PREDICTIONS

Artificial neural networks are one of the prominent techniques that are used in machine learning. A successful neural network requires substantial amount of training data, com-

putational resources and expertise on machine learning which urges small scale companies to outsource their analytics to big service providers. However, outsourcing potentially sensitive data brings a privacy risk to the enterprises. In **Chapter 5**, we propose a protocol to perform neural network predictions under privacy preservation. Our protocol brings together two well-known cryptographic techniques for secure computation: partially homomorphic encryption and secure two-party computation, and computes neural network predictions by switching between the two methods. The hybrid nature of our protocol enables to maintain the accuracy of predictions and to optimize the computation time and bandwidth usage. This chapter is an integral copy of the paper "*SwaNN: Switching among Cryptographic Tools for Privacy-Preserving Neural Network Predictions*" by G. Tillem, B. Bozdemir, and M. Önen which is under review.

CHAPTER 6
PRIVACY-PRESERVING ONLINE BEHAVIOURAL ADVERTISING

In marketing analytics, serving digital advertisements based on the customer's interests benefits both the customers and the product owners. However, data collected from the customers for online behavioral advertising creates concerns over the privacy of the customers. In **Chapter 6**, we propose two protocols for privacy preserving online behavioral advertising which combines machine learning techniques with cryptographic mechanisms. Our first protocol, uses homomorphic encryption to match the user profiles with the right advertisement. Our second protocol improves the performance of the first protocol using a secret sharing scheme which distributes computations between multiple advertising companies. This chapter is an integral copy of the papers "*AHEad: Privacy-preserving Online Behavioural Advertising using Homomorphic Encryption*" by L. Helsloot, G. Tillem, and Z. Erkin in *IEEE Workshop on Information Forensics and Security, WIFS 2017* (pp. 1-6) (2017) and "*BAdASS: Preserving Privacy in Behavioural Advertising with Applied Secret Sharing*" by L. Helsloot, G. Tillem, and Z. Erkin in *JoWUA* (pp.23-41) (2019).

CHAPTER 7
DISCUSSION

In **Chapter 7**, we summarize our solutions and evaluate the contributions of the thesis. We discuss what has been achieved with our proposals and what are the open problems for the future directions of research in the field of confidential data analytics-as-a-service.

1.5.2. LIST OF EXCLUDED PUBLICATIONS

The following is the list publications that are published during the Ph.D. studies but not included in this thesis since they are not directly related to confidential DAaaS.

1. Sheikh Alishahi, M. , **Tillem, G.**, Erkin, Z., Zannone, N., *Privacy-Preserving Multi-Party Access Control*, in *ACM CCS Workshop on Privacy in the Electronic Society* (in press).

2. Nandakumar, L., **Tillem, G.**, Erkin, Z., Keviczky, T., *Protecting the Grid Topology and User Consumption Patterns during State Estimation in Smart Grids based on Data Obfuscation*, in *8th DACH+ Conference on Energy Informatics* (in press).

3. Setia, P. K., **Tillem, G.**, Erkin, Z., *Private Data Aggregation in Decentralized Networks*, in *7th International Istanbul Smart Grids and Cities Congress, ICSG 2019* (2019) pp.76–80.

4. Hoogervorst, R., Zhang, Y., **Tillem, G.**, Erkin, Z., Verver, S., *Solving bin-packing problems under privacy preservation: possibilities and trade-offs*, Information Sciences **500**, 203–216 (2019).

5. Helsloot, L., **Tillem, G.**, Erkin, Z., *BAdASS: Preserving Privacy in Behavioural Advertising with Applied Secret Sharing*, in *12th International Conference on Provable Security, ProvSec 2018* (2018) pp.397–405.

6. Helsloot, L., **Tillem, G.**, Erkin, Z., *Privacy concerns and protection measures in online behavioural advertising*, in 38^{th} *WIC Symposium on Information Theory in the Benelux* (2017) pp.89–96.

7. **Tillem, G.**, Erkin, Z., Lagendijk, R. L., *Privacy-Preserving Alpha Algorithm for Software Analysis*, in 37^{th} *WIC Symposium on Information Theory in the Benelux* (2016) pp.136–143.

1.5.3. ABOUT THE THESIS

Each technical chapter of the thesis (apart from **Chapter 2**) includes at least one publication which are referenced at the beginning of the chapter. We preserve the integral copy of the publications with minor changes. The chapters are independent from each other so they can be read without reading the previous chapters. As a consequence, there might be overlapping parts and similarities in the introduction, preliminaries, and related work sections. Also, due to the same reason, terminology and notation might vary across publications and chapters.

REFERENCES

[1] M. Belfiore, *How 10 industries are using big data to win big,* (2016).

[2] C. Petrov, *Big data statistics 2019,* (2019).

[3] H. Oza, *Here's how big data is changing the world,* (2018).

[4] L. Columbus, *Big data analytics adoption soared in the enterprise in 2018,* (2018).

[5] D. Fogarty and P. C. Bell, *Should you outsource analytics?* MIT Sloan Management Review **55**, 41 (2014).

[6] D. Matthews, *The pros and cons of in-house and outsourced research,* (2015).

[7] C. A. Ardagna, P. Ceravolo, and E. Damiani, *Big data analytics as-a-service: Issues and challenges,* in *2016 IEEE International Conference on Big Data, BigData 2016, Washington DC, USA, December 5-8, 2016* (2016) pp. 3638–3644.

[8] T. Baer, *Big data 2019: Cloud redefines the database and machine learning runs it,* (2019).

[9] M. zur Muehlen and R. Shapiro, *Business process analytics,* in *Handbook on Business Process Management 2, Strategic Alignment, Governance, People and Culture, 2nd Ed.* (2015) pp. 243–263.

[10] M. Jiandani, *Business process analytics – what and why?* (2015).

[11] W. M. P. van der Aalst, *Process Mining - Data Science in Action, Second Edition* (Springer, 2016).

[12] M. Kerremans, *Market Guide for Process Mining,* Tech. Rep. (Gartner, 2019).

[13] D. C. Ciresan, U. Meier, and J. Schmidhuber, *Multi-column deep neural networks for image classification,* CoRR **abs/1202.2745**, 20 (2012).

[14] S. Ruan, J. O. Wobbrock, K. Liou, A. Y. Ng, and J. A. Landay, *Comparing speech and keyboard text entry for short messages in two languages on touchscreen phones,* IMWUT **1**, 159:1 (2017).

[15] N. Jones, *Computer science: The learning machines,* Nature: Computer Science **505**, 146 (2014).

[16] S. Ransbotham, D. Kiron, P. Gerbert, and M. Reeves, *Reshaping business with artificial intelligence: Closing the gap between ambition and action,* MIT Sloan Management Review **59** (2017).

[17] Google, *Cloud machine learning engine,* .

[18] Amazon, *Machine learning on aws,* .

[19] IBM, *Ibm watson machine learning,* .

[20] Research and Markets, *Machine learning as a service (mlaas) market - growth, trends and forecast (2019 - 2024),* (2019).

[21] G. McGovern and J. Quelch, *Outsourcing marketing,* (2005).

[22] C. M. Bishop, *Pattern recognition and machine learning, 5th Edition*, Information science and statistics (Springer, 2007).

[23] J. Schmidhuber, *Deep learning in neural networks: An overview,* Neural Networks **61**, 85 (2015).

[24] S. Rendle, *Factorization machines,* in *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010* (2010) pp. 995–1000.

[25] C. Pemberton, *Key findings from gartner marketing analytics survey 2018,* (2018).

[26] Z. Xiao and Y. Xiao, *Security and privacy in cloud computing,* IEEE Communications Surveys and Tutorials **15**, 843 (2013).

[27] Internal Audit 360°, *Ftc hits facebook with $5 billion fine, huge compliance orders,* (2019).

[28] S. Rodriguez, *Here are the scandals and other incidents that have sent facebook's share price tanking in 2018,* (2018).

[29] Reputation Affairs, *The facebook-cambridge analytica scandal - part 2,* (2018).

[30] European Commission, *Eu data protection rules,* (2019).

[31] M. Gibbons, *Impact of the New GDPR Directive on Outsourcing Arrangements*, Tech. Rep. (Wavestone, 2018).

[32] EU GDPR.ORG, *Gdpr key changes,* (2019).

[33] J. Marinas, *Gdpr and its impact on outsourcing,* (2018).

[34] The DDC Group, *GDPR and Outsourcing - Busting the Myths*, Tech. Rep. (The DDC Group, 2018).

[35] C. Tankard, *What the GDPR means for businesses,* Network Security **2016**, 5 (2016).

[36] Information Commissioner's Office, *Guide to the general data protection regulation (gdpr),* (2019).

[37] Z. Shan, K. Ren, M. Blanton, and C. Wang, *Practical secure computation outsourcing: A survey,* ACM Comput. Surv. **51**, 31:1 (2018).

[38] J. Domingo-Ferrer, O. Farràs, J. Ribes-González, and D. Sánchez, *Privacy-preserving cloud computing on sensitive data: A survey of methods, products and challenges,* Computer Communications **140-141**, 38 (2019).

[39] *Advanced encryption standard,* in *Encyclopedia of Cryptography and Security, 2nd Ed.* (2011) p. 24.

[40] D. X. Song, D. A. Wagner, and A. Perrig, *Practical techniques for searches on encrypted data,* in *2000 IEEE Symposium on Security and Privacy, Berkeley, California, USA, May 14-17, 2000* (2000) pp. 44–55.

[41] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, *Order-preserving encryption for numeric data,* in *Proceedings of the ACM SIGMOD International Conference on Management of Data, Paris, France, June 13-18, 2004* (2004) pp. 563–574.

[42] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, *Private information retrieval,* in *36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, USA, 23-25 October 1995* (1995) pp. 41–50.

[43] O. Goldreich, *Towards a theory of software protection and simulation by oblivious rams,* in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA* (1987) pp. 182–194.

[44] T. E. Gamal, *A public key cryptosystem and a signature scheme based on discrete logarithms,* IEEE Trans. Information Theory **31**, 469 (1985).

[45] C. Gentry, *Fully homomorphic encryption using ideal lattices,* in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009* (2009) pp. 169–178.

[46] A. C. Yao, *How to generate and exchange secrets (extended abstract),* in *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986* (1986) pp. 162–167.

[47] O. Goldreich, S. Micali, and A. Wigderson, *How to play any mental game or A completeness theorem for protocols with honest majority,* in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA* (1987) pp. 218–229.

[48] R. L. Rivest, A. Shamir, and L. M. Adleman, *A method for obtaining digital signatures and public-key cryptosystems,* Commun. ACM **21**, 120 (1978).

[49] D. Boneh and M. K. Franklin, *Identity-based encryption from the weil pairing,* in *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings* (2001) pp. 213–229.

[50] V. Goyal, O. Pandey, A. Sahai, and B. Waters, *Attribute-based encryption for fine-grained access control of encrypted data,* in *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Iooctober 30 - November 3, 2006* (2006) pp. 89–98.

[51] D. Boneh, A. Sahai, and B. Waters, *Functional encryption: Definitions and challenges,* in *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings* (2011) pp. 253–273.

[52] W. M. P. van der Aalst, T. Weijters, and L. Maruster, *Workflow mining: Discovering process models from event logs,* IEEE Trans. Knowl. Data Eng. **16**, 1128 (2004).

2

CONFIDENTIAL DATA ANALYTICS-AS-A-SERVICE

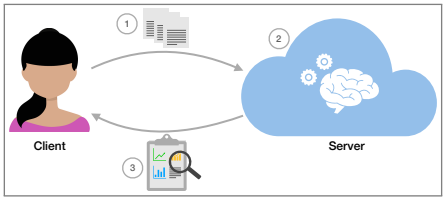
2.1. INTRODUCTION

Confidential data analytics-as-a-service requires to meet several specifications that guarantee the protection of data used for analytics and the analytics results. As introduced in Chapter 1, these specifications are input/output privacy, correctness, and efficiency. A protocol that satisfies these specifications can be designed in several ways depending on the number of parties involved in computations, adversarial behavior, and the cryptographic technique used for the protection of data. In this chapter, we provide the preliminary knowledge on the possible scenarios, adversarial behavior, and cryptographic techniques used for confidential DAaaS. We continue with the presentation of common analytics types used in DAaaS and summarize state-of-the-art solutions that aim to achieve confidentiality in analytics outsourcing. We conclude the chapter with a discussion of open issues and challenges in the existing literature that are addressed in the succeeding chapters of this thesis.

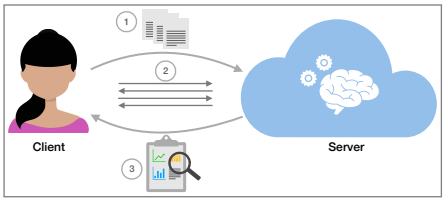
2.2. PRELIMINARIES

2.2.1. SCENARIOS

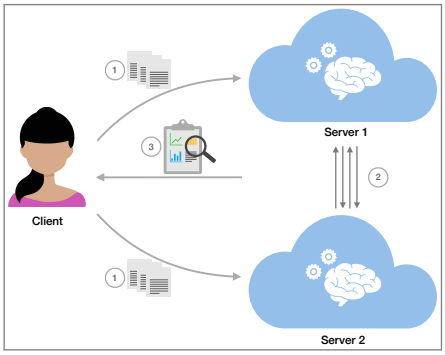
The computation of outsourced analytics by the analytics outsourcing company and the service provider company can be performed in different scenarios concerning the computational capabilities of the outsourcing company and the flexibility of operations. Below we describe three possible scenarios in confidential data analytics-as-a-service, which are a *standalone server scenario*, *client-server scenario*, and a *non-colluding servers scenario*. Figure 2.1 illustrates the interactions between the analytics outsourcing company and the service provider company for each scenario. In the figure and also in descriptions, we refer to the analytics outsourcing company as the client and the service provider company as the server.



(a) Standalone server scenario



(b) Client-server scenario



(c) Non-colluding servers scenario

Figure 2.1: Commonly used scenarios in confidential data analytics-as-a-service.

STANDALONE SERVER SCENARIO

The trivial scenario in confidential data analytics-as-a-service is outsourcing all the analytics computation to the server, which is illustrated in Figure 2.1a. In this scenario, the client is responsible for the preparation of inputs and also deciding what information is sensitive and what information can be shared with the server publicly. The server is responsible to perform the computations on protected data and to return the results to the client under protection. The standalone server scenario is the desired scenario for outsourced computation since it delegates all the computations to the server and does not require any computational power from the client.

CLIENT-SERVER SCENARIO

An alternative to the standalone server scenario is using a client-server scenario, where some of the computations are delegated to the client, as illustrated in Figure 2.1b. This scenario is feasible if the client has adequate computational resources but does not have the knowledge for data analytics. The client outsources the data analytics tasks to the server but helps the server to compute some intermediary operations in the protocol. These operations are usually the ones that are required to be performed on unprotected data. Thus, in the intermediary steps of computation, the client decrypts the protected data, performs the necessary operations, sends the computation result under protection back to the server.

NON-COLLUDING SERVERS SCENARIO

The client-server scenario is a convenient setting for confidential data analytics-as-a-service as soon as the client has adequate computational capabilities. However, especially for the small scale enterprises, having computational resources is not always a realistic assumption. Overcoming the computational cost on the client-side is possible

with delegating the computations to two (or more) servers, as illustrated in Figure 2.1c. In this scenario, the client distributes the protected data among the independent servers according to some predefined rules. The servers collaboratively run the computations and return the result of data analytics to the client under privacy preservation. Since the computations are delegated to two servers, where the client has less control over the computation process, a non-collusion assumption between the servers is important to guarantee security. Accordingly, the servers should not collude during the computation, because otherwise, they can reveal partial or complete information about the input. The collusion among servers can be prevented by law, conflicting interests, or physical means [1].

2.2.2. ADVERSARIAL BEHAVIOR

In the design of a confidential data analytics protocol, it is crucial to determine the adversarial behavior before choosing a data protection mechanism. In the secure computation, an adversary can behave in three possible ways:

- a ***semi-honest adversary*** (a.k.a. *honest-but-curious* or *passive adversary*) follows the computation steps without any deviation. However, he observes the input, output, and intermediary messages to retrieve additional information that should remain private [2]. Although it is considered as a weaker adversarial model, a semi-honest adversarial setting is useful in the cases where the parties trust each other but do not want to leak any information beyond the public knowledge.

- a ***malicious adversary*** (a.k.a. *active adversary*) can arbitrarily deviate from the computation specification [2]. Achieving security against malicious adversaries is desired since the adversarial behavior is stronger. However, the performance overhead of the protocols that are secure against malicious adversaries makes them less practical compared to the alternatives under the semi-honest model.

- a ***covert adversary*** is an intermediate adversarial model which is proposed against the weak security of the semi-honest model and the inefficiency of the malicious model [2]. A covert adversary may behave maliciously. However, if he does so, it will be caught by the honest parties with some probability.

In confidential DAaaS, an adversary can be internal adversary, i.e., the client or the server, or an external adversary. In this thesis, we focus on the protection against the internal adversaries since the external adversaries can be prevented by other means such as physical measures or firewalls. The internal adversary in our protocols is a semi-honest adversary that does not deviate from protocol specification but curious to get more information by observing messages.

2.2.3. CRYPTOGRAPHIC TECHNIQUES

In confidential data analytics, depending on the purpose of computations different cryptographic techniques can be used. In Chapter 1, we list these functionalities as secure search, secure computation and access control. In this thesis, our goal is to perform secure computation for outsourced analytics. Therefore, in the following, we explain

two techniques for secure computation which are homomorphic encryption and secure multiparty computation.

HOMOMORPHIC ENCRYPTION

In cryptography, homomorphism is a property of encryption which allows to perform a certain computation on the encrypted text without decrypting the text. Formally, given the plaintext set M and ciphertext set C, an encryption scheme with encryption function $E(\cdot)$ is homomorphic if

$$\forall m_1, m_2 \in M, \quad E(m_1 \oplus m_2) = E(m_1) \odot E(m_2) \tag{2.1}$$

can be computed directly without intermediate decryptions by a single party [3]. The concept of homomorphism is first introduced by Rivest et al. [4] as privacy homomorphisms, and since then many cryptographic schemes that support this property are proposed. While the early proposals are limited to perform only one type of arithmetic operation, which is either an addition or multiplication, the recent proposals for homomorphic encryption allow both additions and multiplications. Regarding the number of operations allowed on the ciphertext, homomorphic cryptosystems can be divided into three categories:

- **Partially homomorphic encryption** allows one type of arithmetic operation (i.e. either addition or multiplication) on ciphertext which can be performed unlimited number of times [5]. Depending on the allowed arithmetic operation, the cryptosystem can be additively homomorphic such as Paillier [6], Goldwassser - Micali [7], Damgård-Jurik [8] or DGK [9] cryptosystems, or it can be multiplicatively homomorphic as in RSA [10] and ElGamal [11] cryptosystems.
- **Somewhat homomorphic encryption** allows some type of arithmetic operations on ciphertext which can be performed only limited number of times [5]. The type of arithmetic operations allowed in somewhat homomorphic schemes are usually unlimited number of additions and limited number of multiplications. The mutliplications are limited since the expansion in ciphertext makes the schemes impractical. BGN [12], IP [13], Polly Cracker [14], and SYY [15] are some examples of somewhat homomorphic cryptosystems.
- **Fully homomorphic encryption** allows any type of arithmetic operations (i.e. both additions and multiplications) on ciphertext which can be performed unlimited number of times [5]. Gentry's proposal in [16] is the first cryptosystem that achieves full homomorphism. The existing fully homomorphic cryptosystems are divided into three groups with respect to the underlying mathematical constructions as the schemes based on ideal lattices [16, 17], the schemes based on the hardness of finding an approximate GCD of large integers [18], and the schemes based on the learning with errors and ring learning with errors problems [19].

The main bottleneck in the deployment of homomorphic encryption schemes is the computation and memory overhead. Since Gentry's proposal of fully homomorphic encryption, several techniques proposed to make fully homomorphic encryption practical that includes optimizations through the use of GPUs or FPGAs in the implementation [20]. Despite the optimizations, with the increasing number of multiplications, the

performance of fully homomorphic schemes are still far from practical. Using a somewhat homomorphic cryptosystem can improve computation performance when it is possible to limit the number of operations. Partially homomorphic cryptosystems offer more efficient solutions concerning computation cost but their functionalities are limited. Several works proposed a switching mechanism for partially homomorphic cryptosystems that combines an additively homomorphic cryptosystem and a multiplicatively homomorphic cryptosystem with an interactive switching phase in a two-party setting [21, 22]. Although the underlying cryptosystems are efficient in homomorphic operations, the proposed switching mechanisms are expensive in computation and communication costs such that the repetitive usage of switching mechanism creates a significant performance overhead.

SECURE MULTIPARTY COMPUTATION

Secure multiparty computation (MPC) enables multiple parties to jointly evaluate a function on their private inputs without revealing anything other than the output such that

$$f(x_1, x_2, \cdots, x_n) = y, \tag{2.2}$$

where f is the function on private inputs x_i $i \in [1, n]$ of n parties, and y is the output of the computation. In secure computation, privacy and correctness are the two fundamental requirements [2]. Guaranteeing privacy in MPC means the protocol does not leak any information beyond the intended information, i.e. the output. On the other hand, correctness in MPC guarantees each party to receive the correct output.

The concept of secure multiparty computation is formally introduced by Yao [23] in a two-party setting as a solution to millionaires' problem, where two parties try to decide who is richer without leaking their actual wealth to each other. Goldreich et al. [24] generalized Yao's proposal to a multiparty setting and provided the feasibility results for MPC for semi-honest adversaries and for malicious adversaries with honest majority.

Considering the fact that any polynomial time function can be represented as a combinatorial circuit of polynomial size [25], MPC protocols aim to design circuits that can secretly evaluate the function f on the private inputs of the parties. There are three different circuit types that are used in secure computation which are Yao's garbled circuits, Boolean circuits, and arithmetic circuits.

- **Garbled circuits:** As the first proposal of MPC, Yao's garbled circuits provide efficient constructions for two-party secure computation [23, 25]. The two party in the computations are called a garbler and an evaluator. In the computation of the function f, the *garbler* encrypts the function to a garbled circuit along with his input while the *evaluator* evaluates the circuit with his input. The result of evaluation reveals the correct output which corresponds to the inputs of the garbler and the evaluator. Garbled circuits use semantically secure symmetric encryption and oblivious transfer as building blocks [25]. XOR and AND gates are primitive gates of garbled circuits that can be used to construct the function f.

- **Boolean circuits:** GMW protocol in [24] generalizes garbled circuits to a multiparty setting using Boolean sharing that describes the circuit as a binary circuit. In Boolean sharing, each party holds an XOR-based secret share for each input

wire [25, 26]. Since the circuit is binary, for an integer input value, the operations should be performed on each bit of the input individually. Similar to garbled circuits, oblivious transfer is used as a building block in computations, and XOR and AND gates are the primitive gates of computations [25].

- **Arithmetic circuits:** An alternative to GMW protocol is using arithmetic sharing which evaluates the circuit on additive secret shares that are created on integers instead of binary values [26, 27]. The primitive gates of arithmetic circuits are addition and multiplication gates. Additions can be computed locally by each party while the computation of multiplication requires to generate multiplication triplets using Beaver's method [28]. In the generation of the multiplication triplets, homomorphic encryption [29] or oblivious transfer [30, 31] can be used as a building block.

Compared to homomorphic encryption, secure multiparty computation provides efficient solutions in computation time since the operations are performed on smaller input sizes. Furthermore, MPC schemes are more flexible in computations since XOR and AND gates are sufficient to form any complex logic function. This flexibility allows for a structural design approach such that one can compile a function into an MPC circuit while with homomorphic encryption, this is less obvious. The drawback of the MPC schemes is its interactive nature which might cause high bandwidth usage in protocol execution with the growing circuit size. However, since the introduction of MPC, many optimizations are proposed to further improve the performance of MPC circuits regarding their computation and communication cost [32–36]. Furthermore, several tools such as FairplayMP [37], Sharemind [38], ABY [26], PICCO [39] which provide implementation of different circuit types are proposed to design complex functions for secure computation protocols.

Despite the optimizations in the design of secure multiparty computation and homomorphic encryption schemes, both techniques still incur overhead on computation or communication cost compared to the original performance of computations. Applying these techniques directly to securely compute data analytics may not result in a practical solution. The problem of practicality necessitates the design of tailored protocols for confidential data analytics that optimizes the overhead in performance while guaranteeing data protection using homomorphic encryption or secure multiparty computation.

2.3. APPLICATIONS OF DATA ANALYTICS-AS-A-SERVICE

In this section, we describe three analytics applications that are preferred by companies in data analytics-as-a-service. The applications we choose are process analytics, machine learning, and marketing analytics. For each application, we also provide a summary of existing work that aims to achieve privacy preservation in the computation of the analytics.

2.3.1. PROCESS ANALYTICS

Process analytics enables organizations to gain insights about the effectiveness of their organizational processes through the observation of the process performance and com-

pliance. The analysis on the process behavior can be performed to understand what happened in the past, what is happening right now, and what might happen in the future [40]. Process mining is one technique that provides an *ex-post* analysis of process behavior by automatically constructing models that explain the behavior of previously recorded event logs [41].

Process mining starts from an event log which keeps record of the events performed during the execution of a process. An event log consists of several cases which are process instances that are handled in the current execution of the process. Each case has a trace of events that keeps information about the current operation performed. Each event has several attributes such as the activity label of the operation performed, the resource who performed the event, the timestamp of the time of occurrence [41]. Table 2.1 illustrates an example event log for handling compensation requests [42]. Each case in the example event log represents an individual who appealed for the compensation. Every step performed for the individual is recorded in the log as an event that can be identified with a unique event id. Every event stores information about the activity performed in that event, the timestamp of the event, the resource who performed the event, and the cost associated with the event.

Given an event log, process mining can be performed in three steps which are *process discovery*, *conformance checking*, and *process enhancement*.

Process Discovery: The first step of process mining is to discover a process model from raw event logs to visualize the process behaviour seen in the log [42]. Regarding the attributes in the event log, the discovery is performed on different perspectives such as *control-flow perspective* which is based on the ordering of activities, *organizational perspective* which is based on the relations of resources, and *case perspective* which is based on the properties of cases [41]. The common practice in process mining is to use control-flow perspective to discover the process models. The model can be represented using different modelling languages such as BPMN [43], Petri nets [44], or EPC [45].

Figure 2.2 illustrates a process model for the example compensation request handling event log using Petri nets. In the model process starts with a *register request*. Once the *register request* is executed, three possible actions, i.e. *examine thoroughly*, *examine casually*, and *check ticket*, can be performed. Among these actions, there is a choice relation between *examine thoroughly* and *examine casually* such that if one of them is executed the other one is disabled. On the other hand, *check ticket* can be performed concurrently with any of the two examine actions. The action *decide* is performed only if both *check ticket* and one of the examine actions are executed. The decision leads to three outcomes, which are *pay compensation*, *reject request*, and *reinitiate request*. If the compensation is paid or the request is rejected, then the process ends. However, if further processing is needed, then the request is forwarded back to ticket check and examine actions.

There exist several algorithms for the discovery of processes from event logs. Alpha algorithm [46] is one of the earliest discovery algorithms which provides a baseline for the successor discovery algorithms. The Alpha algorithm generates process models by discovering sequential patterns, more specifically, direct-succession relations, in the event logs. However, the algorithm is not practical since it cannot handle noise and in-

Table 2.1: An example event log for handling compensation requests [42].

Case id	Event id	Attributes			
		activity	timestamp	resource	cost
1	35654423	register request	30-12-2010:11.02	Pete	50
	35654424	examine thoroughly	31-12-2010:10.06	Sue	400
	35654425	check ticket	05-01-2011:15.12	Mike	100
	35654426	decide	06-01-2011:11.18	Sara	200
	35654427	reject request	07-01-2011:14.24	Pete	200
2	35654483	register request	30-12-2010:11.32	Mike	50
	35654485	check ticket	30-12-2010:12.12	Mike	100
	35654487	examine casually	30-12-2010:14.16	Pete	400
	35654488	decide	05-01-2011:11.22	Sara	200
	35654489	pay compensation	08-01-2011:12.05	Ellen	200
3	35654521	register request	30-12-2010:14.32	Pete	50
	35654522	examine casually	30-12-2010:15.06	Mike	400
	35654524	check ticket	30-12-2010:16.34	Ellen	100
	35654525	decide	06-01-2011:09.18	Sara	200
	35654526	reinitiate request	06-01-2011:12.18	Sara	200
	35654527	examine thoroughly	06-01-2011:13.06	Sean	400
	35654530	check ticket	08-01-2011:11.43	Pete	100
	35654531	decide	09-01-2011:09.55	Sara	200
	35654533	pay compensation	15-01-2011:10.45	Ellen	200
4	35654641	register request	06-01-2011:15.02	Pete	50
	35654643	check ticket	07-01-2011:12.06	Mike	100
	35654644	examine thoroughly	08-01-2011:14.43	Sean	400
	35654647	reject request	12-01-2011:15.44	Ellen	200

completeness [42]. Despite its impracticality, the idea of discovery of direct sequential patterns in the Alpha algorithm inspired more advanced discovery algorithms such as heuristic miner [47, 48] and inductive miner [49–51]. Using the direct-succession relations as the basis, these algorithms provide more robust solutions for process discovery by overcoming the drawbacks of the former algorithms.

Conformance Checking: The second step of process mining is conformance checking where the behaviour observed in the event log is compared with the expected behaviour in the process model. The process model used for the conformance checking can be generated beforehand using a process discovery algorithm or it can be designed manually. The goal of conformance checking is to find the commonalities and discrepancies between the observed behaviour and the expected behaviour, which is useful for performance analysis, compliance check, and internal auditing [42].

Checking the conformance of processes and event logs can be performed in sev-

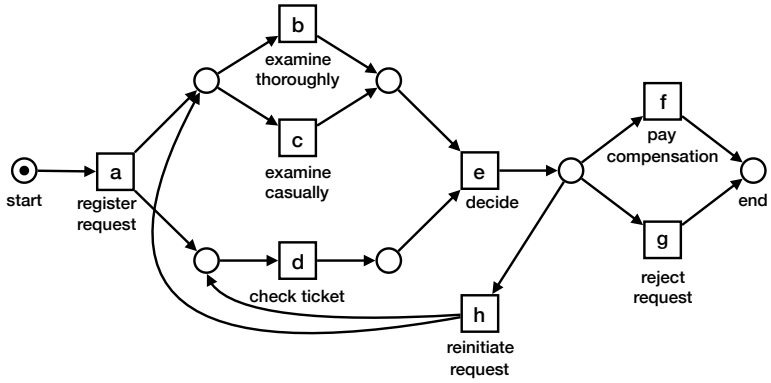


Figure 2.2: Compensation request handling process modelled with Petri net modelling language [42].

eral ways such as token-based replay [52], comparison of footprints [42], and alignments [53]. Alignments offers a robust solution for conformance checking by providing detailed diagnostics at the case level which can be aggregated into the process level. Furthermore, alignments are not specific to a modelling language, rather they can be used for any notation. The idea behind the alignments is to compare each trace with the process model and assign a penalty cost for each deviation in the comparison. The alignment with the lowest cost gives the optimal alignment between the process and the trace.

Figure 2.3 gives three example alignments between the case id 4 in the example log in Table 2.1 and the process model in Figure 2.2. The upper part of the alignment represents the moves, i.e. actions, performed on the log while the lower part is for the moves on the model. In the case of a misalignment between the log and the model, the move is only performed on the model, and the move on the log is skipped with $\gg$, or vice-versa. There is no fixed cost function for misalignments, rather different cost functions can be defined depending on the severity and the likelihood of deviation [42]. For the example alignments in Figure 2.3, considering that a cost of 1 is assigned for each deviation, the alignment cost for γ_1 becomes 1, for γ_2 becomes 9, and for γ_3 becomes 5. Among the example alignments, γ_1 is the optimal alignment with the lowest cost of alignment.

$$\gamma_1 = \begin{array}{|c|c|c|c|c|} \hline a & b & d & \gg & g \\ \hline a & b & d & e & g \\ \hline \end{array} \qquad \gamma_2 = \begin{array}{|c|c|c|c|c|c|c|c|c|} \hline a & b & d & g & \gg & \gg & \gg & \gg & \gg \\ \hline \gg & \gg & \gg & \gg & a & c & d & e & f \\ \hline \end{array}$$

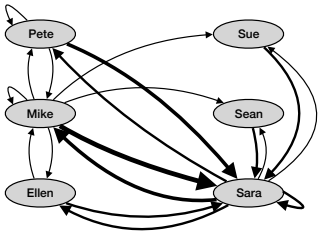
$$\gamma_3 = \begin{array}{|c|c|c|c|c|c|c|} \hline a & b & \gg & d & g & \gg & \gg \\ \hline a & \gg & c & d & \gg & e & g \\ \hline \end{array}$$

Figure 2.3: Example alignments between the case id 4 and the model in Figure 2.2.

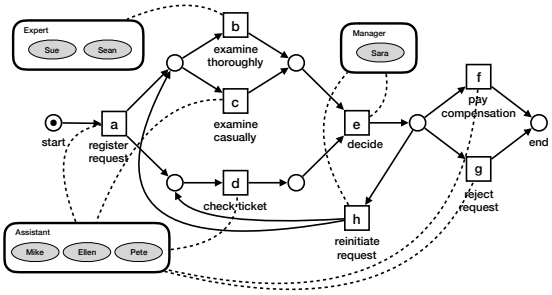
Process Enhancement: The third step of process mining is enhancement which aims to repair or extend an existing process model with other information recorded in the event log [42]. The reparation of the process model is necessary when the observed behaviour of the system differentiates from the modelled behaviour. Extension, on the other hand, adds a new perspective to the process model such as organizational perspective or time perspective.

A common type of process enhancement is extending the process model with the resource information, which is known as organizational mining [54]. It is useful to observe organizational structure, work distribution, and patterns. Organizational mining can be represented as a social network which shows the interactions among different resources with different frequencies, as in Figure 2.4a, or it can be integrated to the process model to show the organizational structure as in Figure 2.4b.

Mining time perspective is another type of process enhancement which helps to observe the duration of activities, detect bottlenecks, monitor resource utilization, and estimate remaining execution time for the ongoing cases [42].



(a) Social network for the event log in Table 2.1 which shows the handover of the work on the individual level [42].



(b) Process model extended with the organizational perspective [42].

Figure 2.4: Organizational mining.

PRESERVING CONFIDENTIALITY IN PROCESS ANALYTICS

Process mining is an emerging discipline which is empowered with the recent omnipresence of the event logs and the rise of data mining. Therefore, the main focus of the research on process mining so far has become development and deployment of well-established algorithms that can maturely perform process mining task. Despite the considerable involvement of data in the process mining algorithms, privacy and security aspects of process mining has not been discussed in the existing literature thoroughly.

In the general domain of business process management, majority of research is focused on proposing semantics or a language for security and privacy requirements in the design of business processes. While some researchers aim to cover the requirements in a broader sense such that confidentiality, integrity, and availability are all covered with the proposed semantics [55–58], some target only the protection of personal data and propose privacy requirements for business processes [59–61]. Another line of the research

focuses on the detection and measurement of information flow leakages in business processes [62–65].

Below we discuss the existing works that aims to achieve protection of confidentiality in process discovery and conformance checking. To the best of our knowledge, privacy protection in process enhancement has not been addressed in any existing work.

Preserving Confidentiality in Process Discovery Existing works that aim to preserve confidentiality in process discovery provide efficient solutions in computation time. Yet, they suffer from the protection of inputs and outputs. In [66], Burattin et al. proposes to use encryption for the protection of event logs such that categorical attributes like activity name or resource name are encrypted with the symmetric cryptosystem AES [67] and the numerical attributes like timestamp and cost are encrypted using the additively homomorphic cryptosystem of Paillier [6]. The encrypted values are not used for any process mining operation. The authors only report the encryption and decryption time for event logs and discuss the possibility of using such encrypted logs in process mining operations such as process discovery. As an initial attempt to preserve privacy in process mining, Burattin et al.'s proposal, unfortunately, cannot achieve privacy. In the encryption of categorical attributes, AES used in a deterministic way which means different encryptions of the same value results in the same ciphertext. Despite the authors claim that this encryption mechanism anonymizes the dataset, the deterministic nature of the encryption rather works as a pseudonym generator for the categorical attributes. It does not assure the input and output privacy requirements of confidential data analytics. Although, the values are hidden, it is possible to perform frequency analysis and background knowledge attacks on the encrypted values.

Rafiei et al. [68] extends [66] by proposing a framework for confidential process mining and designing a process discovery protocol to discover directly follows relations from encrypted data. Similar to Burattin et al., they also propose to use deterministic encryption for the protection of categorical attributes. To improve privacy protection, they remove the connection between each event and the case id that corresponds to the event. While this approach prevents leaking the control flow of a single case, it still does not protect against the information leakage by frequency analysis and susceptible to background knowledge attacks.

The underlying challenge in process discovery, which is mining the "directly follows relations", is also related to sequential pattern mining techniques of data mining [69]. There exist several studies that aims to achieve confidentiality in sequential pattern mining using homomorphic encryption and secure multiparty computation such as [70–74]. Unfortunately, these mechanisms cannot be directly applied to process discovery since the directly follows relations is loosened in these techniques by removing the requirement of being direct successors. Therefore, we do not provide a detailed analysis of confidential the sequential pattern mining solutions in this thesis.

Preserving Confidentiality in Conformance Checking To the best of our knowledge there is no work that specifically focuses on the privacy aspects of conformance checking in process mining domain. However, there exist techniques that are similar to conformance checking in the broader application of business process management which

are log auditing and business process matching.

Guanciale et al. [75] proposes a protocol for log auditing under privacy preservation. In their setting, two mutually distrustful parties such that one owns the event log and the other one owns the business process, want to compute the mismatches between the log and the process. They propose to use secure multiparty computation protocols in [36] which are based on the universally composable arithmetic black box functionality that allows users to store, retrieve and process values securely. Their proposal first transforms a process model to a finite state automata and then checks whether the alphabet in the log matches the alphabet of the finite state machine. Despite its efficiency, the solution proposed in [75] is not applicable for conformance checking since it does not measure the cost of misalignment. Rather the proposed solutions returns a binary value that indicates a match or mismatch between the log and the process.

In [76], Gurov et al. uses the same arithmetic black box functionality to compute business process matching. In business process matching, the goal is to measure similarity of two business processes, rather than a comparison of an event log and a business process. The matching performed by [76] is not a Boolean operation, rather they aim to measure the misalignment between the two models. However, their misalignment measurement technique differs from the classical techniques of conformance checking for measuring the cost of misalignment. The authors propose to use a graph similarity metric [77] to measure the the misalignment which may not be applied to conformance checking setting.

2.3.2. MACHINE LEARNING

Many organizations are using machine learning techniques to improve their services and increase their profit. Machine learning helps organizations in several tasks such as decision making, forecasting, and recommenders. Artificial neural networks are one popular technique in machine learning technology which aim to solve a classification problem by correctly assigning a label to a new observation through the analysis of previous observations [78]. Artificial neural networks, as other machine learning techniques, operate in two phases:

- a **training phase**, where a neural network model is trained through observation of a training set. The quality of the trained networks is determined by its classification accuracy on the training set. However, finding the right parameters that achieve high quality classification is not trivial. Computation of the network parameters require to solve an optimization problem. Gradient descent is a method to solve the optimization problem in neural networks that is used together with a backward propagation algorithm to find a high quality neural network [79]. Performing training is an expensive operation with respect to computation time especially with the increasing training dataset size. With the advent of new technologies such as the usage of GPUs in computation, the training times can be reduced to a feasible range.

- a **prediction phase**, where a new observation is assigned a class label using the trained neural network. Compared to the training phase, prediction phase is less

costly since the computation is only a feedforward operation that requires to operate on the neural network once.

An artificial neural network consists of three main layers which are an input layer, where the input is feed to the network, an output layer where the result of classification is revealed, and the hidden layers, where the main classification operation is computed. Figure 2.5 illustrates the main layers of artificial neural networks.

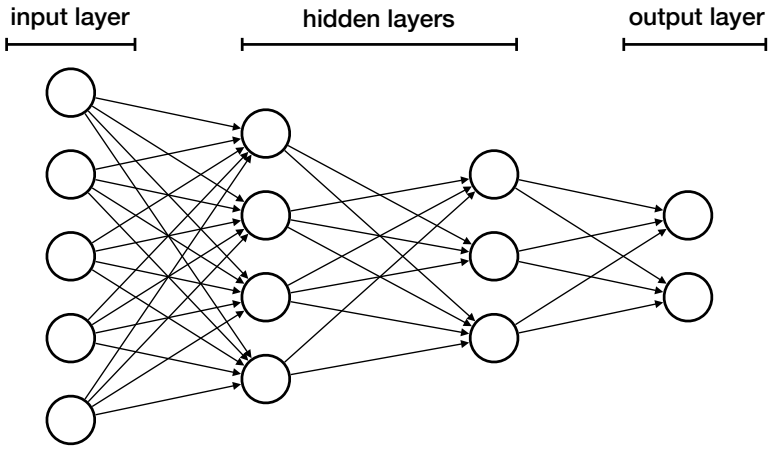


Figure 2.5: An example artificial neural network structure with input, output, and hidden layers.

With respect to the purpose of classification, there exists different types of artificial neural networks such as convolutional neural networks, recurrent neural networks, or feedforward neural networks. Convolutional neural networks are one common type of neural networks which are used for image processing. There exists several kinds of hidden layers that are commonly used in convolutional neural networks. Below we briefly describe some of them.

- **Fully connected layer** connects each neuron in the previous layer to each neuron in the current layer with a certain weight. The connection between neurons can be computed using matrix multiplication. Establishing connection among each neurons causes an expansion in the network size.
- **Convolutional layer** aims to reduce the expansion caused by fully connected layer using convolutional filters in image processing. Convolutional filter can be applied on the input by performing dot products.
- **Activation layer** computes a nonlinear function to improve classification accuracy which is especially useful for complex classification problems [78]. The activation function is computed on each neuron of the previous layer individually, so it does not cause an expansion or reduction in the number of neurons. Sigmoid, rectified linear unit (ReLU), and hyperbolic tangent are some well-known activation functions used in neural networks.

- **Pooling layer** is another layer that is used to reduce the number of neurons in the network. The reduction is performed either by max pooling which outputs the maximum value in a subset of the input or by average pooling which outputs the average of the values in a subset of the input.

PRESERVING CONFIDENTIALITY IN MACHINE LEARNING

The expensive nature of machine learning algorithms triggered outsourcing of machine learning applications by enterprises to cloud servers. With the increased usage of outsourcing in machine learning, privacy of the outsourced data and analytics also gained importance. Despite the research on privacy-preserving neural network computations is recent, it is growing fast and a handful of solutions that aim to achieve privacy in neural networks have been proposed. Below we discuss several prominent proposals that focus on protection of confidentiality in neural networks.

The research on confidential neural network trainings is very limited due to the expensive nature of training algorithms. To the best of our knowledge, SecureML by Mohassel et al [80] is the only work that performs neural network trainings under protection. They use secure two-party computation based on garbled circuits and Boolean circuits to compute the hidden layers. Since secure computation on nonlinear activation is not trivial, only ReLU and x^2 functions are used in the activation layer. As stated previously, due to the expensive nature of training, the secure computations can only be performed on small network sizes.

On contrary to the high costs in training phase, there exists several proposals that computes prediction phase of neural networks under privacy-preservation efficiently. The existing solutions for private neural network predictions can be divided into three categories such that the solutions based on fully homomorphic encryption, the solutions based on secure multiparty computation, and hybrid solutions that combine homomorphic encryption with secure multiparty computation.

CryptoNets [81] presents one of the first algorithms that compute private neural network predictions using fully homomorphic encryption. The authors use the standalone server scenario, where all computations are delegated to the cloud server. In such a scenario, the flexibility of the operations are limited to the types of operations allowed by the underlying homomorphic cryptosystem. The authors propose to use a homomorphic encryption scheme which requires to know the number of arithmetic operations to be used beforehand [82]. Using this cryptosystem, convolutional and fully connected layers can be computed smoothly, but the computation of activation and pooling layers require some approximations. The authors propose to approximate the activation function by using x^2 as the activation function. For the pooling layer, they introduce *scale mean pooling* which only aggregates the values in a subset of the input without performing division. Using approximations induces the problem of loss of accuracy in predictions. The experiment results show that using CryptoNets a prediction accuracy of 98.95% can be achieved on MNIST dataset [83] while the state-of-the-art accuracy rate for the same dataset is 99.79% [84]. Despite the loss in accuracy, computations in CryptoNets do not incur any computation cost on the client-side. However, the cost of performing a single prediction takes around 300 seconds using CryptoNets. An advantage of using CryptoNets in neural network computations is the ability to batch computations

such that in a single computation, the prediction for 4096 images can be computed. This is advantageous when the prediction dataset that the client holds is large.

An alternative solution to CryptoNets is MiniONN by Liu et al. [84] which computes predictions using secure two-party computation. MiniONN uses a client-server setting, where the computations are shared between the client and the server. The secure two-party computation operations are performed on arithmetic circuits and garbled circuits. The flexibility of garbled circuits enables MiniONN to work on more accurate activation functions such that the protocol achieves an accuracy of 99.31% on MNIST dataset compared to 98.95% accuracy of CryptoNets on the same dataset. Furthermore, MiniONN can perform a single prediction on MNIST dataset using the same network structure in CryptoNets in 1.28 seconds. The bottleneck of the MiniONN is the high bandwidth usage (~48 MB for the sample network) that is caused by the interactive nature of secure two-party computation.

A third approach in the computation of private neural network operations is using a hybrid mechanism that combines homomorphic encryption with secure multiparty computation. The motivation for a hybrid solution derives from high bandwidth usage of secure multiparty computation and high computation cost and degraded accuracy of fully homomorphic encryption. Gazelle [85] is one such protocol that combines an additively homomorphic encryption system with garbled circuits. The homomorphic cryptosystem used in Gazelle, packed additively homomorphic encryption (PAHE), is a special system proposed by the authors which can perform fast matrix and vector multiplications and convolutions [85]. Thanks to the combination of PAHE with garbled circuits, Gazelle can outperform CryptoNets and MiniONN both in computation cost and bandwidth usage such that a single prediction with Gazelle can be computed in 30 ms with 0.5 MB bandwidth usage. The disadvantage of Gazelle lies in the practicality of the protocol. The homomorphic cryptosystem proposed in the paper is specific to Gazelle such that it is complex and there is no implementation available. This problem makes the reproducibility of the proposed solution impractical.

2.3.3. MARKETING ANALYTICS

An important application of marketing analytics is online advertisements which enable companies to serve their advertisements to targeted users based on their browsing behaviour. However, delivering the advertisements to the customers require to collaborate with the publishers. Online advertisements can be delivered to the customers by using real-time bidding mechanism of buying and selling advertisements [86]. Using the real-time bidding mechanism, the companies can have a better control on their advertisement budget and on the pages and the end users that their advertisements are shown.

In the online behavioral advertisement ecosystem, a company's advertisement is delivered to a user when the users interest matches to the content of the advertisement. Computing the probability of an end-user is interested in the advertisement of a company is possible using several machine learning techniques such as logistic regression or factorization machines.

- **Logistic regression** is one technique to compute the likelihood of a user clicking an advertisement. Given the user profile x as a feature vector and the vector of

model parameters w, logistic regression estimates the binary output (click or no click) $\hat{y}$ using the sigmoid function as follows:

$$\hat{y} = \frac{1}{1 + e^{-w^T x}}. \tag{2.3}$$

To improve the accuracy of prediction, the model parameters w are updated once the actual binary click y is observed. The stochastic gradient descent algorithm can be used to update model parameters [87]. Despite its simplicity and efficiency, logistic regression cannot handle the interactions between different features in user profile which might affect the prediction quality.

- **Factorization machines** are an alternative to logistic regression for the prediction of user interest by considering the interactions among different features. Factorization machines can perform predictions efficiently by reducing the features into a low-dimensional space with the use of feature interactions [88].

PRESERVING CONFIDENTIALITY IN MARKETING ANALYTICS

Different from the other analytics types, in online advertisements, the data that companies perform analytics are collected through the web history of customers and the protection of data directly relates to the protection of customer privacy. For this purpose, several solutions proposed to protect privacy of end users.

One group of solutions aim to prevent tracking of users using ad-blocking mechanisms that blocks requests of advertisers [89]. The blocking tools are installed on the end users machine and does not require any cooperation with publishers or advertisers. However, not all blocking tools guarantee blocking tracking, rather they prevent showing the advertisements to the end users. Using ad blockers is not desired by publishers and companies since companies cannot advertise their products and publishers lose their revenue for offering free content [90].

Anonymizing user's profile using several obfuscation and anonymization techniques is another line of research in private online advertising [91–93]. While this solution benefits the privacy of users, it is not desirable for companies which aim to target their advertisements to users since the obfuscation and anonymization methods add noise to data and produce misleading prediction results for advertising companies [92].

Several solutions uses cryptographic techniques to protect confidentiality of user data while enabling the companies to serve their advertisements under privacy preservation. Juels [94] proposes a scheme which uses private information retrieval (PIR) to deliver the advertisement to the user without leaking information about which advertisement is delivered. The proposed solution is a client-server scenario where the client, end user, computes the advertisement that matches to his/her profile locally. Instead of a single server, the solution proposes to use a group of servers such that the number of honest servers are above a threshold. To anonymize the user who requested the advertisement, a mix network is used. The main bottleneck of Juels' scheme is the computational overhead of PIR approach combined with mix networks that is not applicable for the real-time nature of online advertising [95].

Backes et al. [95] proposes another scheme, ObliviAd, which uses Oblivious RAM (ORAM) instead of PIR and combines it with secure coprocessor. In a client-server mo-

del, using ORAM assures that the advertisement accessed by the client cannot be observed by the server and secure coprocessor protects the client's information from the server. However, the advertising model proposed in ObliviAd does not match with the existing real time bidding model of advertisements. Furthermore, the use of secure coprocessor may not be feasible without increasing the cost beyond the point of profitability [96, 97].

Adnostic [98] is another scheme that uses cryptographic mechanisms to guarantee the privacy of users by protecting the advertisements viewed by the user. It uses a voting system based on homomorphic encryption such that for a set of advertisements downloaded to user's system, it reports the binary information of advertisement (viewed/ not viewed) to the server in an encrypted form. The proposed scheme needs a trusted third party that should decrypt the aggregated statistic of the advertisements to report it to the advertisers. The requirement of downloading the advertisements in advance makes the scheme unsuitable for real time bidding mechanism [97].

2.4. OPEN ISSUES AND CHALLENGES

In this chapter, we provided the preliminary knowledge to design protocols for confidential data analytics-as-a-service applications and described three analytics applications along with their existing privacy-preserving solutions. Considering the existing literature, we clearly see that there is a need for the assurance of confidentiality in process analytics, machine learning, and marketing analytics when the algorithms of these applications performed in an outsourced setting. In this thesis, we propose several protocols that aim to achieve confidentiality in process discovery, conformance checking, neural network predictions, and online behavioral advertisement. We use secure multiparty computation and homomorphic encryption to protect and process data in our protocols. Depending on the computational capabilities of the analytics outsourcing companies, we use either a client-server scenario or a non-colluding servers scenario in our protocols. All of our protocols achieve security against semi-honest adversaries. In each protocol, we aim to achieve the following requirements:

- **Privacy:** In the delivery of inputs to the service provider company and the delivery of outputs to the analytics outsourcing company, and during the computation of analytics the privacy of data should be assured.

- **Accuracy:** The result of secure computations should be correct. If achieving correctness is not possible due to the limitations of the underlying secure computation mechanism, the goal should be to maximize the accuracy of the computations such that it is similar to the expected original results.

- **Efficiency:** The cryptographic mechanism applied for the analytics tasks should not degrade the performance of original computations drastically. The overhead of secure computation should be minimized.

Among these requirements, the privacy of input and output can be verified with the use of provably secure cryptographic mechanisms for secure computation. Therefore,

our main goal in the following chapters is to achieve accuracy and efficiency in the design of confidential data analytics-as-a-service protocols. There exists several challenges that we should address in achieving accuracy and efficiency:

- **Loss of precision:** The analytics types described in this chapter uses small numbers which are not necessarily integers. However, the cryptographic mechanisms for secure computation usually work on integer values. Therefore, in the design of our protocols minimizing the loss of precision in transformation from real numbers to integers incurs a challenge on the accuracy of computations.
- **Nonlinear functions:** Both fully homomorphic encryption and secure multiparty computation can perform arithmetic computations. However, computing nonlinear functions using these secure computation techniques is not trivial. To maximize the accuracy of computations, we need protocols that can compute or approximate nonlinear functions.
- **Trade-off between computation cost and bandwidth usage:** As it is clear in the existing works on secure computation, optimizing computation cost results in a degraded bandwidth usage, or a lower bandwidth usage incurs high computation cost. Feasibility of both computation and communication cost is important for outsourced computation. Therefore, balancing the computation cost and the bandwidth usage is a challenge in the efficiency of our proposals.

In the following chapters, we try to overcome these challenges by designing privacy-preserving protocols that are based on cryptographic techniques. We aim to propose protocols that are clear, understandable, and reproducible by the research community for fair validation and comparison of our protocols with future research attempts.

REFERENCES

[1] S. Kamara, P. Mohassel, and M. Raykova, *Outsourcing multi-party computation,* IACR Cryptology ePrint Archive **2011**, 272 (2011).

[2] C. Hazay and Y. Lindell, *Efficient Secure Two-Party Protocols - Techniques and Constructions,* Information Security and Cryptography (Springer, 2010).

[3] C. Fontaine and F. Galand, *A survey of homomorphic encryption for nonspecialists,* EURASIP J. Information Security **2007** (2007), 10.1155/2007/13801.

[4] R. L. Rivest, L. Adleman, and M. L. Dertouzos, *On data banks and privacy homomorphisms,* Foundations of secure computation **4**, 169 (1978).

[5] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, *A survey on homomorphic encryption schemes: Theory and implementation,* ACM Comput. Surv. **51**, 79:1 (2018).

[6] P. Paillier, *Public-key cryptosystems based on composite degree residuosity classes,* in *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding* (1999) pp. 223–238.

[7] S. Goldwasser and S. Micali, *Probabilistic encryption and how to play mental poker keeping secret all partial information,* in *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA* (1982) pp. 365–377.

[8] I. Damgård and M. Jurik, *A generalisation, a simplification and some applications of paillier's probabilistic public-key system,* in *Public Key Cryptography, 4th International Workshop on Practice and Theory in Public Key Cryptography, PKC 2001, Cheju Island, Korea, February 13-15, 2001, Proceedings* (2001) pp. 119–136.

[9] I. Damgård, M. Geisler, and M. Krøigaard, *Efficient and secure comparison for online auctions,* in *Information Security and Privacy, 12th Australasian Conference, ACISP 2007, Townsville, Australia, July 2-4, 2007, Proceedings* (2007) pp. 416–430.

[10] R. L. Rivest, A. Shamir, and L. M. Adleman, *A method for obtaining digital signatures and public-key cryptosystems,* Commun. ACM **21**, 120 (1978).

[11] T. E. Gamal, *A public key cryptosystem and a signature scheme based on discrete logarithms,* IEEE Trans. Information Theory **31**, 469 (1985).

[12] D. Boneh, E. Goh, and K. Nissim, *Evaluating 2-dnf formulas on ciphertexts,* in *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings* (2005) pp. 325–341.

[13] Y. Ishai and A. Paskin, *Evaluating branching programs on encrypted data,* in *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings* (2007) pp. 575–594.

[14] M. Fellows and N. Koblitz, *Combinatorial cryptosystems galore!* Contemporary Mathematics **168**, 51 (1994).

[15] T. Sander, A. L. Young, and M. Yung, *Non-interactive cryptocomputing for nc^1*, in *40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA* (1999) pp. 554–567.

[16] C. Gentry, *Fully homomorphic encryption using ideal lattices,* in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009* (2009) pp. 169–178.

[17] N. P. Smart and F. Vercauteren, *Fully homomorphic encryption with relatively small key and ciphertext sizes,* in *Public Key Cryptography - PKC 2010, 13th International Conference on Practice and Theory in Public Key Cryptography, Paris, France, May 26-28, 2010. Proceedings* (2010) pp. 420–443.

[18] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, *Fully homomorphic encryption over the integers,* in *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings* (2010) pp. 24–43.

[19] Z. Brakerski and V. Vaikuntanathan, *Fully homomorphic encryption from ring-lwe and security for key dependent messages,* in *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings* (2011) pp. 505–524.

[20] C. Moore, M. O'Neill, E. O'Sullivan, Y. Doröz, and B. Sunar, *Practical homomorphic encryption: A survey,* in *IEEE International Symposium on Circuits and Systemss, ISCAS 2014, Melbourne, Victoria, Australia, June 1-5, 2014* (2014) pp. 2792–2795.

[21] G. Couteau, T. Peters, and D. Pointcheval, *Encryption switching protocols,* in *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I* (2016) pp. 308–338.

[22] G. Castagnos, L. Imbert, and F. Laguillaumie, *Encryption switching protocols revisited: Switching modulo p,* in *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I* (2017) pp. 255–287.

[23] A. C. Yao, *Protocols for secure computations (extended abstract),* in *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982* (1982) pp. 160–164.

[24] O. Goldreich, S. Micali, and A. Wigderson, *How to play any mental game or A completeness theorem for protocols with honest majority,* in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA* (1987) pp. 218–229.

[25] Y. Lindell and B. Pinkas, *Secure multiparty computation for privacy-preserving data mining,* J. Priv. Confidentiality **1** (2009), 10.29012/jpc.v1i1.566.

[26] D. Demmler, T. Schneider, and M. Zohner, *ABY - A framework for efficient mixed-protocol secure two-party computation,* in *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015* (2015).

[27] D. Chaum, C. Crépeau, and I. Damgård, *Multiparty unconditionally secure protocols (extended abstract),* in *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA* (1988) pp. 11–19.

[28] D. Beaver, *Efficient multiparty protocols using circuit randomization,* in *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings* (1991) pp. 420–432.

[29] M. J. Atallah, M. Bykova, J. Li, K. B. Frikken, and M. Topkara, *Private collaborative forecasting and benchmarking,* in *Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society, WPES 2004, Washington, DC, USA, October 28, 2004* (2004) pp. 103–114.

[30] N. Gilboa, *Two party RSA key generation,* in *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings* (1999) pp. 116–129.

[31] J. Bringer, H. Chabanne, M. Favre, A. Patey, T. Schneider, and M. Zohner, *GSHADE: faster privacy-preserving distance computation and biometric identification,* in *ACM Information Hiding and Multimedia Security Workshop, IH&MMSec '14, Salzburg, Austria, June 11-13, 2014* (2014) pp. 187–198.

[32] G. Asharov, Y. Lindell, T. Schneider, and M. Zohner, *More efficient oblivious transfer and extensions for faster secure computation,* in *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013* (2013) pp. 535–548.

[33] M. Naor, B. Pinkas, and R. Sumner, *Privacy preserving auctions and mechanism design,* in *EC* (ACM, 1999) pp. 129–139.

[34] B. Pinkas, T. Schneider, N. P. Smart, and S. C. Williams, *Secure two-party computation is practical,* in *ASIACRYPT*, Lecture Notes in Computer Science, Vol. 5912 (Springer, 2009) pp. 250–267.

[35] R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin, *Efficient multiparty computations secure against an adaptive adversary,* in *EUROCRYPT*, Lecture Notes in Computer Science, Vol. 1592 (Springer, 1999) pp. 311–326.

[36] I. Damgård and J. B. Nielsen, *Universally composable efficient multiparty computation from threshold homomorphic encryption,* in *CRYPTO*, Lecture Notes in Computer Science, Vol. 2729 (Springer, 2003) pp. 247–264.

[37] A. Ben-David, N. Nisan, and B. Pinkas, *Fairplaymp: a system for secure multi-party computation,* in *ACM Conference on Computer and Communications Security* (ACM, 2008) pp. 257–266.

[38] D. Bogdanov, S. Laur, and J. Willemson, *Sharemind: A framework for fast privacy-preserving computations,* in *ESORICS*, Lecture Notes in Computer Science, Vol. 5283 (Springer, 2008) pp. 192–206.

[39] Y. Zhang, A. Steele, and M. Blanton, *PICCO: a general-purpose compiler for private distributed computation,* in *ACM Conference on Computer and Communications Security* (ACM, 2013) pp. 813–826.

[40] M. zur Muehlen and R. Shapiro, *Business process analytics,* in *Handbook on Business Process Management (2)*, International Handbooks on Information Systems (Springer, 2015) pp. 243–263.

[41] W. M. P. van der Aalst, H. A. Reijers, A. J. M. M. Weijters, B. F. van Dongen, A. K. A. de Medeiros, M. Song, and H. M. W. Verbeek, *Business process mining: An industrial application,* Inf. Syst. **32**, 713 (2007).

[42] W. M. P. van der Aalst, *Process Mining - Data Science in Action, Second Edition* (Springer, 2016).

[43] M. von Rosing, S. White, F. Cummins, and H. de Man, *Business process model and notation - BPMN,* in *The Complete Business Process Handbook, Vol. I* (Morgan Kaufmann/Elsevier, 2015) pp. 429–453.

[44] W. Reisig and G. Rozenberg, eds., *Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, the volumes are based on the Advanced Course on Petri Nets, held in Dagstuhl, September 1996*, Lecture Notes in Computer Science, Vol. 1491 (Springer, 1998).

[45] W. M. P. van der Aalst, *Formalization and verification of event-driven process chains,* Information & Software Technology **41**, 639 (1999).

[46] W. M. P. van der Aalst, T. Weijters, and L. Maruster, *Workflow mining: Discovering process models from event logs,* IEEE Trans. Knowl. Data Eng. **16**, 1128 (2004).

[47] A. J. M. M. Weijters and J. T. S. Ribeiro, *Flexible heuristics miner (FHM),* in *CIDM* (IEEE, 2011) pp. 310–317.

[48] A. J. M. M. Weijters and W. M. P. van der Aalst, *Rediscovering workflow models from event-based data using little thumb,* Integrated Computer-Aided Engineering **10**, 151 (2003).

[49] S. J. J. Leemans, D. Fahland, and W. M. P. van der Aalst, *Discovering block-structured process models from event logs - A constructive approach,* in *Petri Nets*, Lecture Notes in Computer Science, Vol. 7927 (Springer, 2013) pp. 311–329.

[50] S. J. J. Leemans, D. Fahland, and W. M. P. van der Aalst, *Discovering block-structured process models from event logs containing infrequent behaviour,* in *Business Process Management Workshops*, Lecture Notes in Business Information Processing, Vol. 171 (Springer, 2013) pp. 66–78.

[51] S. J. J. Leemans, D. Fahland, and W. M. P. van der Aalst, *Discovering block-structured process models from incomplete event logs,* in *Petri Nets*, Lecture Notes in Computer Science, Vol. 8489 (Springer, 2014) pp. 91–110.

[52] A. Rozinat and W. M. P. van der Aalst, *Conformance checking of processes based on monitoring real behavior,* Inf. Syst. **33**, 64 (2008).

[53] W. M. P. van der Aalst, A. Adriansyah, and B. F. van Dongen, *Replaying history on process models for conformance checking and performance analysis,* Wiley Interdiscip. Rev. Data Min. Knowl. Discov. **2**, 182 (2012).

[54] M. Song and W. M. P. van der Aalst, *Towards comprehensive support for organizational mining,* Decision Support Systems **46**, 300 (2008).

[55] J. Müller, J. A. Mülle, S. von Stackelberg, and K. Böhm, *Secure business processes in service-oriented architectures - A requirements analysis,* in *ECOWS* (IEEE Computer Society, 2010) pp. 35–42.

[56] J. A. Mülle, S. von Stackelberg, and K. Böhm, *Modelling and transforming security constraints in privacy-aware business processes,* in *SOCA* (IEEE Computer Society, 2011) pp. 1–4.

[57] C. L. Maines, D. Llewellyn-Jones, S. Tang, and B. Zhou, *A cyber security ontology for bpmn-security extensions,* in *CIT/IUCC/DASC/PICom* (IEEE, 2015) pp. 1756–1763.

[58] O. Altuhhova, R. Matulevicius, and N. Ahmed, *Towards definition of secure business processes,* in *CAiSE Workshops*, Lecture Notes in Business Information Processing, Vol. 112 (Springer, 2012) pp. 1–15.

[59] W. Labda, N. Mehandjiev, and P. Sampaio, *Modeling of privacy-aware business processes in BPMN to protect personal data,* in *SAC* (ACM, 2014) pp. 1399–1405.

[60] A. D. Brucker and I. Hang, *Secure and compliant implementation of business process-driven systems,* in *Business Process Management Workshops*, Lecture Notes in Business Information Processing, Vol. 132 (Springer, 2012) pp. 662–674.

[61] S. Goldner and A. Papproth, *Extending the BPMN syntax for requirements management,* in *BPMN*, Lecture Notes in Business Information Processing, Vol. 95 (Springer, 2011) pp. 142–147.

[62] R. Accorsi and C. Wonnemann, *Strong non-leak guarantees for workflow models,* in *SAC* (ACM, 2011) pp. 308–314.

[63] R. Accorsi, A. Lehmann, and N. Lohmann, *Information leak detection in business process models: Theory, application, and tool support,* Inf. Syst. **47**, 244 (2015).

[64] S. Frau, R. Gorrieri, and C. Ferigato, *Petri net security checker: Structural non-interference at work,* in *Formal Aspects in Security and Trust*, Lecture Notes in Computer Science, Vol. 5491 (Springer, 2008) pp. 210–225.

[65] M. Pettai and P. Laud, *Combining differential privacy and mutual information for analyzing leakages in workflows,* in *POST*, Lecture Notes in Computer Science, Vol. 10204 (Springer, 2017) pp. 298–319.

[66] A. Burattin, M. Conti, and D. Turato, *Toward an anonymous process mining,* in *Fi-Cloud* (IEEE Computer Society, 2015) pp. 58–63.

[67] *Advanced encryption standard,* in *Encyclopedia of Cryptography and Security, 2nd Ed.* (2011) p. 24.

[68] M. Rafiei, L. von Waldthausen, and W. M. P. van der Aalst, *Ensuring confidentiality in process mining,* in *SIMPDA*, CEUR Workshop Proceedings, Vol. 2270 (CEUR-WS.org, 2018) pp. 3–17.

[69] R. Agrawal and R. Srikant, *Mining sequential patterns,* in *ICDE* (IEEE Computer Society, 1995) pp. 3–14.

[70] X. Yi, F. Rao, E. Bertino, and A. Bouguettaya, *Privacy-preserving association rule mining in cloud computing,* in *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '15, Singapore, April 14-17, 2015* (2015) pp. 439–450.

[71] M. Kantarcioglu and C. Clifton, *Privacy-preserving distributed mining of association rules on horizontally partitioned data,* IEEE Trans. Knowl. Data Eng. **16**, 1026 (2004).

[72] T. Tassa, *Secure mining of association rules in horizontally distributed databases,* IEEE Transactions on Knowledge and Data Engineering **26**, 970 (2014).

[73] S. Kim, S. Park, J. Won, and S. Kim, *Privacy preserving data mining of sequential patterns for network traffic data,* Inf. Sci. **178**, 694 (2008).

[74] W. Liu, S. shan Luo, Y. bin Wang, and Z. tao Jiang, *A protocol of secure multi-party multi-data ranking and its application in privacy preserving sequential pattern mining,* in *2011 Fourth International Joint Conference on Computational Sciences and Optimization* (IEEE, 2011) pp. 272–275.

[75] R. Guanciale, D. Gurov, and P. Laud, *Business process engineering and secure multi-party computation,* Applications of Secure Multiparty Computation **13**, 129 (2015).

[76] D. Gurov, P. Laud, and R. Guanciale, *Privacy preserving business process matching,* in *PST* (IEEE Computer Society, 2015) pp. 36–43.

[77] O. Sokolsky, S. Kannan, and I. Lee, *Simulation-based graph similarity,* in *TACAS*, Lecture Notes in Computer Science, Vol. 3920 (Springer, 2006) pp. 426–440.

[78] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff, *Privacy-preserving classification on deep neural network,* IACR Cryptology ePrint Archive **2017**, 35 (2017).

[79] M. A. Nielsen, *Neural networks and deep learning*, Vol. 25.

[80] P. Mohassel and Y. Zhang, *Secureml: A system for scalable privacy-preserving machine learning,* in *IEEE Symposium on Security and Privacy* (IEEE Computer Society, 2017) pp. 19–38.

[81] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. E. Lauter, M. Naehrig, and J. Wernsing, *Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy,* in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016* (JMLR.org, 2016) pp. 201–210.

[82] J. W. Bos, K. E. Lauter, J. Loftus, and M. Naehrig, *Improved security for a ring-based fully homomorphic encryption scheme,* in *IMA Int. Conf.*, Lecture Notes in Computer Science, Vol. 8308 (Springer, 2013) pp. 45–64.

[83] Y. LeCun and C. Cortes, *MNIST handwritten digit database,* (2010).

[84] J. Liu, M. Juuti, Y. Lu, and N. Asokan, *Oblivious neural network predictions via minionn transformations,* in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017* (ACM, 2017) pp. 619–631.

[85] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, *GAZELLE: A low latency framework for secure neural network inference,* in *27th USENIX Security Symposium, USENIX Security 2018, Baltimore,MD, USA, August 15-17, 2018.* (USENIX Association, 2018) pp. 1651–1669.

[86] J. Wang, W. Zhang, and S. Yuan, *Display advertising with real-time bidding (RTB) and behavioural targeting,* Foundations and Trends in Information Retrieval **11**, 297 (2017).

[87] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, S. Chikkerur, D. Liu, M. Wattenberg, A. M. Hrafnkelsson, T. Boulos, and J. Kubica, *Ad click prediction: a view from the trenches,* in *KDD* (ACM, 2013) pp. 1222–1230.

[88] S. Rendle, *Factorization machines,* in *ICDM* (IEEE Computer Society, 2010) pp. 995–1000.

[89] G. Merzdovnik, M. Huber, D. Buhov, N. Nikiforakis, S. Neuner, M. Schmiedecker, and E. R. Weippl, *Block me if you can: A large-scale study of tracker-blocking tools,* in *EuroS&P* (IEEE, 2017) pp. 319–333.

[90] J. Estrada-Jiménez, J. Parra-Arnau, A. Rodríguez-Hoyos, and J. Forné, *Online advertising: Analysis of privacy threats and protection approaches,* Computer Communications **100**, 32 (2017).

[91] M. Degeling and T. Herrmann, *Your interests according to google - A profile-centered analysis for obfuscation of online tracking profiles,* CoRR **abs/1601.06371** (2016).

[92] F. Brunton and H. Nissenbaum, *Vernacular resistance to data collection and analysis: A political theory of obfuscation,* First Monday **16** (2011).

[93] F. Papaodyssefs, C. Iordanou, J. Blackburn, N. Laoutaris, and K. Papagiannaki, *Web identity translator: Behavioral advertising and identity privacy with WIT,* in *HotNets* (ACM, 2015) pp. 3:1–3:7.

[94] A. Juels, *Targeted advertising ... and privacy too,* in *CT-RSA*, Lecture Notes in Computer Science, Vol. 2020 (Springer, 2001) pp. 408–424.

[95] M. Backes, A. Kate, M. Maffei, and K. Pecina, *Obliviad: Provably secure and practical online behavioral advertising,* in *IEEE Symposium on Security and Privacy* (IEEE Computer Society, 2012) pp. 257–271.

[96] M. Green, W. Ladd, and I. Miers, *A protocol for privately reporting ad impressions at scale,* in *ACM Conference on Computer and Communications Security* (ACM, 2016) pp. 1591–1601.

[97] L. J. Helsloot, G. Tillem, and Z. Erkin, *Privacy concerns and protection measures in online behavioural advertising,* (2017).

[98] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas, *Adnostic: Privacy preserving targeted advertising,* in *NDSS* (The Internet Society, 2010).

3

PROCESS DISCOVERY ON ENCRYPTED DATA

The increasing demand for data mining in business intelligence has led to significant growth in the adoption of the data mining-as-a-service paradigm which enables companies to outsource their data mining tasks to a cloud service provider. Process mining is one recently investigated technique for data analysis which enables the discovery of process models from event logs collected during software execution. Despite the popularity of outsourcing in analytics, the companies hesitate to enable the cloud providers' access to their data considering customer privacy and intellectual property. In this chapter, we propose two privacy-preserving protocols for the discovery of process models that assure the privacy for analytics outsourcing companies. Both of our protocols use encryption to protect and process the data using the homomorphic properties of the encryption. Our first protocol, AlphaSec, provides a privacy-preserving solution for a fundamental process discovery algorithm. Our second protocol, PriSM, generalizes AlphaSec to make it applicable for any kind of process discovery algorithm. PriSM uses a novel switching mechanism that allows using both multiplicative and additive homomorphism on the ElGamal cryptosystem. We provide the experiment results for both protocols to show their feasibility in privacy-preserving data analytics applications. To the best of our knowledge, our protocols are the first attempts to achieve process discovery under privacy-preservation with the help of provably secure cryptographic techniques.

This chapter has been published as

- "*Mining Encrypted Software Logs using Alpha Algorithm*" by G. Tillem, Z. Erkin, and R.L. Lagendijk in *SECRYPT*. (pp. 267-274) (2017), which is presented in **Section 3.1**,
- "*Mining Sequential Patterns from Outsourced Data via Encryption Switching*" by G. Tillem, Z. Erkin, and R.L. Lagendijk in *PST*. (pp. 1-10) (2018), which is presented in **Section 3.2**.

3.1. MINING ENCRYPTED SOFTWARE LOGS USING ALPHA ALGORITHM

Software systems have an evolving nature which enables them to respond to the needs of technological advances continuously [1]. While this evolution is advantageous to improve service quality for users, the drawback is growing complexity which complicates the management of software systems [2]. The complication occurs especially in the verification and validation of the system properties. Considering that current systems can reach up to billions of lines of code [3], the classical analysis of software becomes impractical [1]. Overcoming the difficulties of classical approach is possible using model-based analysis techniques. In these techniques, a formal model of a system is generated and the conformance of properties are checked by automated tools to address defects in the design [4].

A common approach in model-based analysis is modeling the system behavior through event logs that contain information about software execution [5]. A promising technique for such an analysis is process mining that aims to *discover*, *monitor*, and *enhance* processes using the information in event logs [6]. The discovery, i.e. *process discovery*, aims to generate a process model from the logs to observe system behavior. Monitoring, or *conformance checking*, compares an existing model with real logs of the same process to conform the real behavior to the expected behavior. Finally, enhancement, i.e. *process enhancement*, improves an existing model with the real event logs, to replay the reality on the existing model.

In every category of process mining, the content of event logs are crucial in the system analysis. The logs may contain information about users (e.g. user id or e-mail), duration of execution, system properties (e.g. memory usage, OS type) or component interactions. Although this information is useful in modelling the behavior, the content might leak sensitive information of owners; user and software company. For a user, sharing sensitive data with third parties may pose a privacy threat. A recent discussion about GHTorrent [7], a platform to monitor and publish GitHub events as dataset, exemplifies such a threat in shared logs. In the dataset user e-mails used to be published since they are already public on GitHub [8]. However, this situation initiated a displeasure when the dataset is used by third companies to send survey e-mails to data owners [8]. The discussion ended by removing personal data from the dataset [8]. Sharing logs is also arguable for software companies regarding the intellectual property and confidential information in logs. [9] shows that it is possible to reverse engineer software logs with process mining. Considering the risk of piracy through reverse engineering [10], the companies are not willing to share information with external parties.

The existing literature on software analysis for security and privacy approaches the problem from several aspects. The studies for the protection of the intellectual property are mostly focus on cryptographic solutions such as code obfuscation [11], watermarking [12] and tamper-proofing [13]. For the protection of user privacy, some studies approach the problem as the privacy of data in testing applications [14, 15] and provide solutions by applying anonymization. Several studies attempt to protect user privacy during log generation by reducing the sensitive information in log reports [16, 17]. Furthermore, the control of information flow between software components is also a con-

cern. [18] and [19] address the problem of controlling sensitive information flow using taint tracking and analysis mechanisms.

While there are many efforts for securing log-based software analysis in the literature, no studies have focused on privacy issues in software analysis with process mining. In this paper, we propose a protocol for privacy-preserving process discovery for software analysis, namely AlphaSec. Thus, we select the alpha algorithm [20] which is a favorable algorithm in understanding the mechanism of discovery with a relatively simple structure.

Our scenario has three parties namely, users, software company (SC) and process miner (PM). The users send the event logs to SC and are not active in the rest of the protocol. PM executes the process discovery protocol on the logs under the supervision of SC. We assume a semi-honest setting where PM and SC do not collude. In order to achieve privacy, we encrypt the logs under a homomorphic cryptosystem. To identify the items in the logs and the relations between them, we use several cryptographic protocols as secure equality checking, secure multiplication and bit decomposition. Furthermore, we use data packing to eliminate the repetition of same operations and to exploit encryption modulus optimally. During the protocol execution, PM and SC are not allowed to directly decrypt the logs. Moreover, the decryptions on intermediate values are secured. In this setting, our protocol guarantees the privacy of data owners. To the best of our knowledge, our paper presents the first protocol for privacy-preserving software analysis with process mining which assures both user and software privacy. Our protocol does not change the original structure of alpha algorithm and it can be adapted to other discovery algorithms with slight modifications. While our proposal adopts well-known cryptographic protocols, it reduces the cost of those protocols significantly by using data packing. We provide computational and communication complexity analysis along with experiments to show the improvement of our protocol.

3.1.1. PRELIMINARIES

In this section we summarize the alpha algorithm and and introduce the cryptographic tools used in our protocol. Table 3.1 summarizes the notation.

THE ALPHA ALGORITHM

The alpha algorithm takes an event log $L = \{\sigma_0, \cdots, \sigma_\tau\}$ as input, where L is a set of traces σ_i such that every σ_i is composed of events $e_{j\sigma_i}$, scans it to find patterns and outputs the result as a Petri net[1] [20]. Moreover, every $e_{j\sigma_i}$ contains several attributes, such as activity, timestamp or resource which determine the perspective of process discovery. Following the common approach in process mining, in this work we assume that activity attribute is used for process discovery, so every $e_{j\sigma_i}$ has only one attribute which is activity.

The algorithm runs in 8 steps [6]. In **Steps 1-3**, the set of activities appeared in L, $T_L \subset T$, and the sets of the first ($T_I \subset T$) and last ($T_O \subset T$) activities are discovered. **Step 4** aims to discover the ordering relations between activities. The ordering is based on *direct succession*, $t_b > t_c$, which means t_c directly follows t_b in σ_i. The direct successions are used to define 3 ordering relations which are 1. *Causality* ($t_b \rightarrow t_c$ or $t_c \leftarrow t_b$): $t_b > t_c$,

[1] A modeling language used in process mining. See [20] for details.

Table 3.1: Explanation of the notation.

Symbol	Explanation
T	Set of activities t_i s.t. $T = \{t_1, t_2, \cdots, t_\Delta\}$
σ_i	A trace with ω_i events s.t. $\sigma_i = \langle e_{1\sigma_i}, \cdots, e_{\omega_i\sigma_i}\rangle$
$e_{j\sigma_i}$	j^{th} event of σ_i, where $1 \leq j \leq \omega_i$ and $1 \leq i \leq \tau$
L	Event log with τ traces, s.t. $L = \{\sigma_0, \cdots, \sigma_\tau\}$
$\otimes$	Secure multiplication operator
$\oplus$	Homomorphic addition operator
$M_{x\times y}$	A matrix M of size $x \times y$
$M_{x,y}$	Index of matrix M in row x and column y
$M_{*,y}$	y^{th} column of matrix M
θ	Compartment size for data packing
N	Plaintext modulus for Paillier cryptosystem
$\mu_{\mathbf{X}}$	Number of packs for the packed array $\mathbf{X}$

but not $t_c > t_b$, 2. *Parallel* $(t_b \,||\, t_c)$: both $t_b > t_c$ and $t_c > t_b$, and 3. *Choice* $(t_b \,\#\, t_c)$: neither $t_b > t_c$ nor $t_c > t_b$. The result of orderings is represented as a footprint matrix. Once the footprint matrix is created, the pairs with causality relation are collected in X_L and in **Step 5** the maximal pairs of X_L are assigned to Y_L. In **Steps 6-7** the set of places P_L and the set of arches, F_L, which connects the elements of P_L are determined. Finally, **Step 8** returns the result $\alpha(L)$ as $(P_L,\ T_L,\ F_L)$.

To illustrate how the alpha algorithm works, we provide a toy example in the following. Let $L = \{\langle a,\ b, e, f\rangle,\ \langle a, b, e, c, d, b, f\rangle,\ \langle a, b, c, e, d, b, f\rangle,\ \langle a, b, c, d, e, b, f\rangle,\ \langle a, e, b, c, d, b, f\rangle\}$ be an event log. The 8 steps of the alpha algorithm for L is:

- $T_L = \{a, b, c, d, e, f\}$, $T_I = \{a\}$, $T_O = \{f\}$.
- $X_L = \{(\{a\}, \{b\}), (\{a\}, \{e\}), (\{b\}, \{c\}), (\{b\}, \{f\}), (\{c\}, \{d\}), (\{d\}, \{b\}), (\{e\}, \{f\}), (\{a, d\}, \{b\}), (\{b\},\ \{c, f\}\}$. See the footprint matrix in Table 3.2 for orderings.

Table 3.2: Footprint matrix for L.

	a	b	c	d	e	f
a	#	→	#	#	→	#
b	←	#	→	←	||	→
c	#	←	#	→	||	#
d	#	→	←	#	||	#
e	←	||	||	||	#	→
f	#	←	#	#	←	#

- $Y_L = \{(\{a\}, \{e\}), (\{c\}, \{d\}), (\{e\}, \{f\}), (\{a, d\}, \{b\}), (\{b\},\ \{c, f\}\}$.
- $P_L = \{i_L, o_L, p_{(\{a\},\{e\})}, p_{(\{c\},\{d\})}, p_{(\{e\},\{f\})}, p_{(\{a,d\},\{b\})}, p_{(\{b\},\{c,f\}}\}$.

- $F_L = \{(i_L, a), (f, o_L), (a, p_{(\{a\},\{e\})}), (p_{(\{a\},\{e\})}, e), (c, p_{(\{c\},\{d\})}), \cdots, (p_{(\{b\},\{c,f\})}, c), (p_{(\{b\},\{c,f\})}, f)\}$.
- The output $\alpha(L) = (P_L, T_L, F_L)$ as in Figure 3.1.

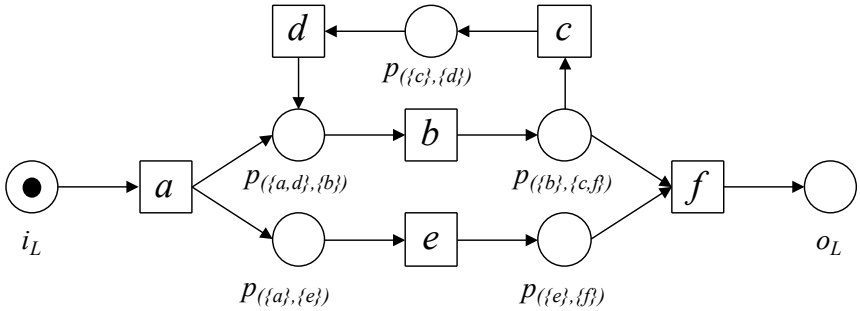


Figure 3.1: The output of the alpha algorithm for the example L as Petri net.

The output of the alpha algorithm is used in conformance checking and process enhancement, to observe the system behavior and to detect the deviations.

PAILLIER CRYPTOSYSTEM

For our protocol we select Paillier cryptosystem [21] for the encryption of L due to its homomorphic property. In Paillier, encryption of a message m modulus $N = p \cdot q$ is performed as $E(m) = g^m \cdot r^N \mod N^2$, where p, q are large primes, $g = N + 1$ and $r \in_R \mathcal{Z}_N^*$. We refer readers to [21] for details of decryption scheme. Paillier cryptosystem enables to perform homomorphic addition on ciphertexts as $E(m_1) \times E(m_2) = E(m_1 + m_2)$. In the rest of the paper, we represent a Paillier ciphertext by $[\cdot]$ and a homomorphic addition by $\oplus$, for the sake of simplicity.

DATA PACKING

In our protocol to eliminate the cost of repeated operations, we use data packing as in [22]. The bit size of inputs in plaintext, determines the compartment size, θ, in packed ciphertext. The number of items in one pack is computed as $\rho = \lfloor \log_2 N/\theta \rfloor$ where $\log_2 N$ is the length of plaintext modulus. Let $[W] = \{[w_0], \cdots, [w_{s-1}]\}$ be an encrypted array of s elements, w_i, we pack $[W]$ into $\mu = \lceil s/\rho \rceil$ ciphertexts such that $[W_{pack}] = \{[W_{pack_0}], \cdots, [W_{pack_{\mu-1}}]\}$ where data packing for every $[W_{pack_t}]$ is performed as

$$[W_{pack_t}] = \sum_{j=0}^{\rho-1} [w_j] \cdot (2^\theta)^j, \text{ s.t. } 0 \le t \le \mu - 1.$$

Using $[W_{pack}]$, we can simultaneously employ homomorphic addition and also reduce the total cost of decryption. In the rest of the paper, we represent data packing as $pack([W], \theta, N)$.

HOMOMORPHIC PROTOCOLS

For encrypted data processing, we use secure equality check [23], secure multiplication [22], and bit decomposition [24] protocols.

Secure Equality Check (SEQ) The common approach to securely check whether $[x] = [y]$ is to check if $[q] = [x - y]$ is 0. One way to test if $[q] = 0$ is to use Hamming distance as in [25]. In our work, we use NEL-I SEQ protocol from [23] that is an efficient version of [25]. We refer reader to [25] and [23] for the details.

Secure Multiplication Protocol (SMP) [22] presents an SMP protocol, where Alice has $[a]$ and $[b]$ and Bob holds the secret key as follows. Alice selects randoms r_a, $r_b \in_R Z_N$, blinds the inputs as $[a'] = [a] \cdot [-r_a]$, $[b'] = [b] \cdot [-r_b]$ and sends $[a']$, $[b']$ to Bob. After decryption, Bob computes $a' \cdot b'$, and sends $[a' \cdot b']$ to Alice. Computing $[a \cdot b] = [a' \cdot b'] \cdot [b]^{r_a} \cdot [a]^{r_b} \cdot [-r_a \cdot r_b]$, Alice gets the encrypted multiplication.

Bit Decomposition (BD) Using BD protocol in [24], Alice and Bob can compute the encrypted bits of an ℓ-bit x as follows. Assume Alice has $[x]$, and Bob holds the secret key. Alice blinds $[x]$ as $[z] = [x - r]$, where $r \in_R \{0,1\}^{\ell+\kappa}$, and sends $[z]$ to Bob. After decryption, Bob sends the least significant ℓ bits of z to Alice in encrypted form. Using $[c_i] = [z_i]^{r_i} \cdot [c_{i-1}]^{r_i} \cdot [z_i \cdot c_{i-1}]$, $[x_i] = [z_i] \cdot [r_i] \cdot [c_{i-1}] \cdot [c_i]^{-2}$, Alice computes the set $\{[x_0], [x_1], \cdots, [x_{\ell-1}]\}$ which is BD of $[x]$.

3.1.2. ALPHASEC: SECURE ALPHA ALGORITHM

In this section, we introduce the privacy-preserving alpha algorithm protocol, namely AlphaSec.

SCENARIO

Our scenario has three parties:

1. ***Software Company (SC)*** is the owner of the software product who holds public and private keys (pk, sk) and stores the encrypted logs.
2. ***Users*** are the users of the software who send the encrypted logs to SC and are not active in the rest.
3. ***Process Miner (PM)*** is a service provider for SC who models the software. PM has the knowledge and resources to perform process mining techniques, thus, SC needs PM's expertize to analyze the software.

Our goal is to minimize the information leakage for users and SC during the protocol execution. Thus, PM must not access the content of encrypted logs and his statistical observations should be restricted. He should not learn the frequencies, but can only observe the ordering relation between two encrypted activities. For instance, for activities a and b, PM can see that $[a] > [b]$ without knowing the values of $[a]$ and $[b]$ and the frequencies of $[a]$, $[b]$ and $[a] > [b]$. On the other hand, SC is only allowed to decrypt the intermediate blinded values and the output of the protocol which contains his own

information. In this setting, our protocol is based on semi-honest security model where PM and SC are non-colluding.

SETUP

In the setup phase, SC generates (pk, sk) and shares pk with PM and users. We assume that SC shares T with PM as $[T] = \{[t_1], \cdots, [t_\Delta]\}$. Furthermore, SC collects $[L] = \{\langle [e_{1\sigma_1}], \cdots [e_{\omega_1\sigma_1}]\rangle, \cdots, \langle [e_{1\sigma_\tau}], \cdots [e_{\omega_\tau\sigma_\tau}]\rangle\}$ from users and shares it with PM to run AlphaSec.

PROCESS MODEL DISCOVERY

AlphaSec protocol focuses on the first 4 steps of the original alpha algorithm, since the sensitive data is processed in these steps. Accordingly, the first task is the discovery of activities T_L, T_I and T_F in encrypted domain, i.e. **Steps 1-3**. The second task is to find the ordering relations, i.e. **Step 4**. Afterwards, a footprint matrix is constructed and **Steps 5-8** of the original algorithm are operated in plaintext. Thus, our protocol is based on 3 subprotocols which are 1. **Secure Activity Discovery**, where the activities are discovered, 2. **Secure Direct Succession Discovery** where the orderings are determined and 3. **Secure Modeling** where the eventual process model is generated.

Algorithm 1 shows how AlphaSec works. When SC requests a process model, in Step 1, PM creates 3 matrices, namely $R_{\Delta\times\Delta}$, $ID_{\Delta\times 1}$ and $FD_{\Delta\times 1}$. While R is used to store direct successions and discovered activities, ID and FD are used to store the initial and final activities. Between Steps 2-4, for each $[\sigma_i]$ of $[L]$, **Secure Activity Discovery** and **Secure Direct Succession Discovery** subprotocols are operated subsequently. After all $[\sigma_i]$s are scanned, a Petri net is generated in Step 5, by **Secure Modelling** subprotocol.

Algorithm 1 AlphaSec

Input: $[L]$, $[T]$

1: R, ID, FD
2: **for all** $[\sigma_i] \in [L]$ **do**
3: (AD^{σ_i}, ID, FD) =**SecureActivityDiscovery**$([\sigma_i])$
4: R = **SecureDirectSuccessionDiscovery(**AD^{σ_i}**)**
5: $\alpha([L])$ = **SecureModelling(**R**,** ID**,** FD**)**

Output: $\alpha([L])$

Secure Activity Discovery The first subprotocol aims to securely discover T_L, T_I and T_O as shown in Algorithm 2. Accordingly, PM collaborates with SC to compare every $[e_{j\sigma_i}]$ with every $[t_m]$ using SEQ and the result is stored in $AD^{\sigma_i}_{\Delta\times\omega_i}$. As showed in Step 3, if $[e_{j\sigma_i}] = [t_m]$, $AD^{\sigma_i}_{m,j}$ is set to [1], else to [0]. Finally, in Step 4, ID and FD are updated with $AD^{\sigma_i}_{*,1}$ and $AD^{\sigma_i}_{*,\omega_i}$, respectively. In Figure 3.2a, we illustrate the procedure for the sample $[L]$.

Since SEQ is an expensive protocol that has to be repeated $\Delta\cdot\omega_i$ times for each σ_i, we use data packing in our protocol. Notice that only a number of intermediate steps of the adopted SEQ protocol [23] can be modified for data packing. We use $pack([e_{j\sigma_i} - t_m], \theta, N)$ as packing function where $\theta = (\lceil \log_2 \Delta \rceil + \kappa)$, $\mu = \Delta\cdot\omega_i/\rho$ and $\rho = \lfloor \log_2 N/\theta \rfloor$.

Algorithm 2 Secure Activity Discovery

Input: $[\sigma_i]$, ID, FD

1: **for all** $[e_{j\sigma_i}] \in [\sigma_i]$ where $1 \leq j \leq \omega_i$ **do**
2: **for all** $[t_m] \in [T]$ where $1 \leq m \leq \Delta$ **do**
3: $AD^{\sigma_i}_{m,j} = ([e_{j\sigma_i}] \stackrel{?}{=} [t_m])\ ?\ [1]\ :\ [0]$
4: $ID = ID \oplus AD^{\sigma_i}_{*,1}$, $FD = FD \oplus AD^{\sigma_i}_{*,\omega_i}$

Output: AD^{σ_i}, ID, FD

	[a]	*[b]*	*[e]*	*[f]*
[a]	[1]	[0]	[0]	[0]
[b]	[0]	[1]	[0]	[0]
[c]	[0]	[0]	[0]	[0]
[d]	[0]	[0]	[0]	[0]
[e]	[0]	[0]	[1]	[0]
[f]	[0]	[0]	[0]	[1]

(a) AD^{σ_1} for σ_1 of L.

	[a]	*[b]*	*[c]*	*[d]*	*[e]*	*[f]*
[a]	[0]	[4]	[0]	[0]	[1]	[0]
[b]	[0]	[0]	[3]	[0]	[2]	[4]
[c]	[0]	[0]	[0]	[3]	[1]	[0]
[d]	[0]	[3]	[0]	[0]	[1]	[0]
[e]	[0]	[2]	[1]	[1]	[0]	[1]
[f]	[0]	[0]	[0]	[0]	[0]	[0]

(b) Final R matrix.

	[a]	*[b]*	*[c]*	*[d]*	*[e]*	*[f]*
[a]	0	1	0	0	1	0
[b]	0	0	1	0	1	1
[c]	0	0	0	1	1	0
[d]	0	1	0	0	1	0
[e]	0	1	1	1	0	1
[f]	0	0	0	0	0	0

(c) Result of zero-check.

	[a]	*[b]*	*[c]*	*[d]*	*[e]*	*[f]*
[a]	#	→	#	#	→	#
[b]	←	#	→	←	||	→
[c]	#	←	#	→	||	#
[d]	#	→	←	#	||	#
[e]	←	||	||	||	#	→
[f]	#	←	#	#	←	#

(d) Footprint matrix.

Figure 3.2: Illustrating AlphaSec protocol on the sample log.

Secure Direct Succession Discovery The next step in AlphaSec is to identify direct successions between activities. To detect subsequent events in $[\sigma_i]$, we merge two subsequent columns of AD^{σ_i} by SMP. Thus, every element in the former column, $AD^{\sigma_i}_{*,j}$ is securely multiplied with every element in the transpose of latter column $(AD^{\sigma_i}_{*,j+1})^T$. Then, the result is added to corresponding index of R.

This subprotocol has two bottlenecks in terms of efficiency. First, the inputs of SMP are encrypted bits, so the plaintext space is not optimally used. Second, for every σ_i SMP protocol runs $\Delta^2 \cdot (\omega_i - 1)$ times. These bottlenecks require us to use data packing. Accordingly, we pack the column $AD^{\sigma_i}_{*,j+1}$ as $pack(AD^{\sigma_i}_{*,j+1}, \theta, N)$, where $\theta = \lceil \log_2 \Gamma \rceil$ and the column $AD^{\sigma_i}_{*,j}$ as $pack(AD^{\sigma_i}_{*,j}, \theta, N)$, where $\theta = \lceil \log_2 \Gamma \rceil \cdot \Delta$ and Γ is the number of events in L. Since the protocol requires to add the result to R, we select a larger compartment size, which is the total number of events in the worst case. The result of SMP is a

packed ciphertext with $\theta = \lceil \log_2 \Gamma \rceil \cdot \Delta$. The number of compartments in one pack and the number of packs are $\rho_1 = \lfloor \log_2 N / \lceil \log_2 \Gamma \rceil \cdot \Delta \rfloor$, $\mu_1 = \Delta \cdot \omega_i / \rho_1$ and $\rho_2 = \lfloor \log_2 N / \lceil \log_2 \Gamma \rceil \rfloor$, $\mu_2 = \Delta \cdot \omega_i / \rho_2$, respectively. In this setting, SMP runs $\mu_1 \cdot \mu_2 \cdot (\omega_i - 1)$ times for every σ_i. In Algorithm 3, we show how to perform secure direct succession discovery with packing. The result of SMP, $mult$, is stored in R_{pack}, whose size is $\mu_1 \cdot \mu_2$.

Algorithm 3 Secure Direct Succession Discovery

Input: AD^{σ_i}

1: **for** $1 \le j \le \omega_i - 1$ **do**
2: $AD^p_1 = pack(AD^{\sigma_i}_{*,j}, \theta, N), AD^p_2 = (AD^{\sigma_i}_{*,j+1}, \theta, N)$
3: **for** $1 \le k \le \mu_1$ **do**
4: **for** $1 \le m \le \mu_2$ **do**
5: $mult = AD^p_{1_k} \otimes AD^p_{2_m}$
6: $R_{\text{pack}_{k,m}} = R_{\text{pack}_{k,m}} \oplus mult$

Output: R_{pack}

After the execution of subprotocol, the result R_{pack} is unpacked using BD to create R. It is important to mention that BD outputs individual bits, but every index of R is a $\lceil \log_2 \Gamma \rceil$-bit integer. Thus, after BD, we perform data packing for every $\lceil \log_2 \Gamma \rceil$ bits to create R. Figure 3.2b shows R matrix for the sample L.

Secure Modelling In the last step of AlphaSec, the output $\alpha([L])$ is generated using R, ID, FD. Here PM needs to know which activity pairs have an ordering relation, but the frequency of the relation should be hidden from him. Thus, we perform a zero-check function on the inputs to observe whether two encrypted activities has an ordering relation, also, whether an activity is first or last activity. For zero-check, PM blinds $R_{i,j}$ with $r \in_R \mathbb{Z}_N$ as $[R'_{i,j}] = [R_{i,j}]^r$ where $1 \le i, j \le \Delta$ and sends $[R'_{i,j}]$ to SC for a secure decryption. If the result of the decryption is non-zero, which means the activity pairs have a direct succession relation, then SC sends 1 and otherwise sends 0 to PM. Hence, PM can only observe the relation between two encrypted activities, but nothing else. Using the result of zero-check, the footprint matrix can be constructed and then the output is generated as in the original alpha algorithm. The only difference is that activities are encrypted and only SC can decrypt them. In Figure 3.2c-3.2d, we illustrate the result of zero-check on R and the footprint matrix, respectively.

3.1.3. PROTOCOL ANALYSIS

In this section, we first provide a security analysis for our protocol, then analyze its computational and communicational complexity and show experimental results. In Table 3.3, we summarize the notation.

SECURITY ANALYSIS

The privacy considerations in our protocol are twofold: user privacy and software company privacy. On one hand, users want to protect their sensitive information from PM and SC. On the other hand, SC wants to protect the intellectual property of his product

Table 3.3: Summary of the notation for complexity analysis.

Notation	Explanation
Γ	Total number of events in L, s.t. $\Gamma = \sum_{i=1}^{\tau} \omega_i$
HAD	Homomorphic addition
HSM	Homomorphic scalar multiplication
ZCF	Zero check function
SEQ	Secure Equality Check
SMP	Secure Multiplication
BD	Bit Decomposition
SAD	Secure Activity Discovery
SDS	Secure Direct Succession Discovery
MD	Secure Modelling

from PM. In the following, we analyze how these concerns are overcome against each party.

Users are not active during protocol execution. They only take part in generation of $[L]$, so they do not have an active adversarial role in our setting.

PM has access to $[L]$ and the results of SEQ, SMP and HAD. The cryptographic protocols are proven to be secure, thus, we assume that PM cannot infer any additional information. Furthermore, to prevent statistical inferences, we hide the frequencies from PM by zero-check. PM can only observe the ordering between two encrypted activities. However, it is not an advantage for PM since the real values are unknown.

SC holds sk and collaborates with PM to operate SEQ and SMP protocols. As the owner of sk, he does not have direct access to $[L]$ to assure user privacy. During SMP, decryption result is blinded, thus, SC cannot infer the original values. For SEQ, we rely on the security of the underlying protocol.

COMPUTATIONAL ANALYSIS

Prior to the analysis of AlphaSec, we analyze the computational complexity of the original alpha algorithm. The operations in the original algorithm are mostly integer or string comparisons which detect distinct activities and the orderings. Thus, T_L, T_I and T_O can be discovered in Γ comparisons. For the discovery of direct successions, every $e_{j\sigma_i}$ can be paired with its successor in Γ operations. Then, the footprint matrix can be generated with at most Δ^2 comparisons.

For the analysis of AlphaSec, we count the number of operations in every subprotocol and illustrate them in Table 3.4 without packing (w/o Packing) and with packing (w/ Packing). Apart from the operations in Table 3.4, Γ and Δ encryptions are performed to encrypt L and T in setup. In AlphaSec, SDS dominates the computations by the quadratic complexity of SMP and HAD. Using data packing, the number of SMP reduces

from Δ^2 to $\lceil(\Delta/\rho_1)\rceil \cdot \lceil(\Delta/\rho_2)\rceil$, where

$$\rho = \lfloor \log_2 N/(\kappa + \lceil \log_2 \Delta \rceil) \rfloor,$$
$$\rho_1 = \lfloor (\log_2 N - \kappa)/(\lceil \log_2 \Gamma \rceil \Delta) \rfloor,$$
$$\rho_2 = \lfloor (\log_2 N - \kappa)/(\lceil \log_2 \Gamma \rceil) \rfloor.$$

Table 3.4: The number of operations performed in AlphaSec.

		w/o Packing	**w/ Packing**
SAD	SEQ	$\Delta\Gamma$	$\lceil \Delta\Gamma/\rho \rceil$
	HAD	$2\cdot(\tau-1)\Delta$	–
SDS	SMP	$\Delta^2(\omega_i-1)\tau$	$\lceil(\Delta/\rho_1)\rceil \lceil(\Delta/\rho_2)\rceil (\omega_i-1)\tau$
	HAD	$\Delta^2(\omega_i-1)\tau$	$\lceil(\Delta/\rho_1)\rceil \lceil(\Delta/\rho_2)\rceil (\omega_i-1)\tau$
	BD	–	$\lceil(\Delta/\rho_1)\rceil \lceil(\Delta/\rho_2)\rceil$
SM	HSM	Δ^2	–
	ZCF	Δ^2	–

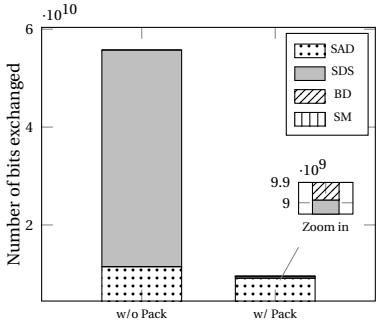
COMMUNICATIONAL ANALYSIS

In Table 3.5, we summarize the communication complexity of AlphaSec in terms of the number of ciphertexts exchanged both for packed and unpacked version. The numbers show that data packing cannot reduce the bandwidth usage for SEQ proportional to the number of packed ciphertext but it reduces the bandwidth usage in intermediate steps. On the other hand, for SMP, the reduction in bandwidth usage is directly proportional to the number of packs.

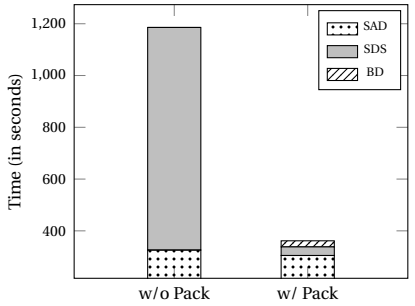
Table 3.5: Bandwidth usage of AlphaSec in terms of the number of exchanged ciphertexts, where $\chi = (\log_2 \log_2 \Delta)$.

	w/o Packing	**w/ Packing**
SEQ	$\Delta\Gamma(3+\lceil \log_2 \Delta \rceil + 2\lceil \chi \rceil)$	$3\Delta\Gamma/\rho + \Delta\Gamma(\lceil \log_2 \Delta \rceil + 2\lceil \chi \rceil)$
SMP	$3\Delta^2(\omega_i-1)\cdot\tau$	$3\lceil\Delta/\rho_1\rceil \lceil\Delta/\rho_2\rceil (\omega_i-1)\tau$
BD	–	$(3(\log_2 N-\kappa)-1)\lceil\Delta/\rho_1\rceil \lceil\Delta/\rho_2\rceil$
ZCF	Δ^2	–

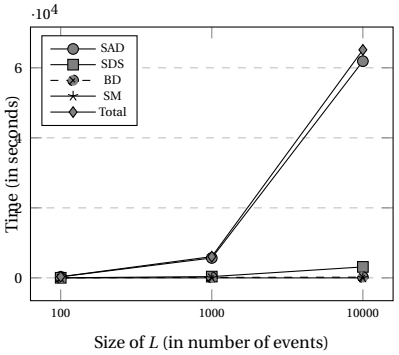
For numerical analysis, we measure the bandwidth usage for a dataset with $\Gamma = 10000$ events, $\Delta = 20$ activities, $\tau = 1000$ traces and $w_i = 10$ with and without packing, where ciphertext size 4096 bits. The comparison results in Figure 3.3a show that data packing can reduce the communication cost significantly. The total improvement in communication cost is 83%, which is mainly based on SDS, where the bandwidth usage of SMP is reduced by a factor of 133. We provide a zoom in to show the communication cost of SDS and BD for w/ Pack, but SM is not visible due to its insignificant cost.



(a) Bandwidth usage of AlphaSec with and without packing.



(b) Performance of AlphaSec in seconds without and with data packing.



(c) Execution time of AlphaSec on different datasets.

Figure 3.3: Evaluating the performance of AlphaSec protocol.

EXPERIMENTS

To measure the real time performance of AlphaSec, we implemented it in C++ with GMP-6.1.2 library. The machine we use runs OSX El Capitan with Intel Core i5 2.7 GHz processor. We choose $\log_2 N = 2048$ for Paillier and $\kappa = 80$ as security parameter. As dataset, we select 3 synthetic datasets (D_1, D_2, D_3) from the event log dataset of IEEE TF on Process Mining[2], where for D_1 $\Gamma = 109$, $\tau = 13$ and $\Delta = 10$, for D_2 $\Gamma = 1,226$, $\tau = 100$ and $\Delta = 16$, and for D_3 $\Gamma = 10696$, $\tau = 1000$ and $\Delta = 20$.

As the first experiment, we measure the effect of packing on performance. Thus, we run AlphaSec on D_1 to compare the timing for SAD, SDS, and BD on packed and unpacked inputs. Since BD is only used when data is packed, we separate it from SDS. Furthermore, we do not include SM in results, since it is same for packed and unpacked data. As the results in Figure 3.3b show applying packing in SDS reduces the computa-

[2] http://data.4tu.nl/repository/collection:event_logs

tion time significantly. The improvement in the computation of SDS is 96% while the total improvement is 71% approximately. On the other hand, SAD is not affected significantly by packing, since it cannot be fully adapted to SEQ.

In the second experiment, we observe the performance on different dataset sizes. Thus we compare the timing of AlphaSec on D_1, D_2, D_3. We run this experiment only on the packed version and measure the time required for SAD, SDS, BD, SM and the total time as illustrated in Figure 3.3c. For D_3 it takes 65133 seconds to run AlphaSec, of which 61885 seconds are spent for SAD, i.e. SEQ. However, performing SDS requires 3135 seconds including BD which takes around 210 seconds. Finally, SM can be performed approximately in 3 seconds.

3.1.4. CONCLUSION

In this paper, we present the first privacy-preserving protocol in process mining for model-based software analysis with the alpha algorithm. The output of our protocol can be used as an input for other process mining techniques such as conformance checking or process enhancement under a privacy-preserving setting. As a first attempt to provide dual privacy for users and SC, we propose a solution based on cryptographic primitives, which provides provable security and privacy. To achieve our goal we use homomorphic encryption along with two-party cryptographic protocols. To reduce the number of operations, we applied data packing on our computations. The performance analyses show that the employment of cryptographic techniques on log analysis provides encouraging results. Furthermore, applying data packing improves the performance significantly.

Although the state-of-the-art process mining techniques are efficient in plaintext domain, our protocol proposes a way to protect sensitive data with additional computational overhead which is promising for the future of this research line. The research challenge is to improve the efficiency of our protocol further by designing custom-tailored cryptographic protocols to replace costly operations such as SEQ and deploying our ideas on more complex process discovery algorithms. With our proposal, we aim to attract the attention of the research community to the privacy aspects of model-based software analysis, which is a distinct and important topic that deserves to be investigated.

3.2. MINING SEQUENTIAL PATTERNS FROM OUTSOURCED DATA VIA ENCRYPTION SWITCHING

Data mining has gained a significant importance in business intelligence with the increasing availability of data collected from information systems. Companies are eager to apply data mining in their businesses to analyze the trends, improve customer satisfaction, and detect problems in their services. However, especially for small-scale companies, obtaining the capabilities and experience for data storage and mining is both challenging and costly [26]. Fortunately, the emergence of data mining as a service paradigm has relieved companies in performing data mining tasks. This paradigm enables a company to outsource its data and data mining tasks to a cloud service provider [27]. It is highly appreciated by companies such that the size of the market is expected to reach $5.9 billion by 2020 [28].

Despite the increasing popularity of data mining-as-a-service, the protection of pri-

vacy sensitive outsourced data is a main concern for companies [27, 29, 30]. The companies hesitate to share their data considering their customers' privacy and intellectual property of their systems [27, 30]. The concern of privacy was triggered further by GDPR in the EU, and the research in the field of privacy-preserving data mining has gained significant importance. A number of works in the field adapt $k-$anonymity [31] as an approach for mining association rules and frequent itemsets [32–35]. Data perturbation and differential privacy are alternatives to k-anonymity for privacy-preserving mining of frequent patterns [36–39]. Moreover, several works adapt differential privacy and data perturbation to achieve privacy in decision tree mining [40, 41], k-means clustering [42], and support vector machines [43]. As an alternative to aforementioned approaches which come with the trade-off of utility and privacy, a number of cryptographic approaches are considered. The cryptographic solutions mostly based on secure multiparty computation [44–49] along with several works based on encryption [50–52]. However, these approaches are usually more expensive in terms of communication and computation, since the operations require mostly two or more parties to mutually perform computations on larger bit sizes.

In this paper, we present a protocol, PriSM (Privacy-preserving Sequential pattern Mining), which aims to mine direct sequential patterns from outsourced data in privacy-preserving manner. We focus on direct sequential pattern mining since it is a commonly used primitive in business process analysis. The ultimate aim of our protocol is to generate a graph from the mined patterns which visualises the behaviour of a business process. Accordingly, the data type we consider here is event logs collected from business enterprise systems. To the best of our knowledge, this problem is studied in two other works by Burattin et al. [53] and Tillem et al. [52]. Burattin et al. [53] present an approach to hide sensitive data in the discovery of process using encryption. While it is an initial attempt in the field, the explanation of approach, its performance, and security are not well-detailed. Tillem et al. [52] have proposed a protocol - AlphaSec - for the discovery of processes under encryption. The protocol adapts an existing discovery algorithm, Alpha algorithm [20], to a privacy-preserving setting. It utilizes expensive two-party protocols based on homomorphic encryption such as secure equality check and secure multiplication. The drawback is that such two-party protocols demands high computation cost along with large number of communication rounds which affects the efficiency of the protocol.

To overcome the drawback of high computation cost in the existing works with formal security guarantees, we design PriSM protocol which is executed between a data owner and a data analyst. The data owner outsources its data to the data analyst under protection. Data analyst runs PriSM protocol to mine direct sequential patterns under supervision of the data owner. The result of mining is delivered to the data owner as a direct sequential patterns graph. In this scenario, to protect sensitive information and at the same time to process the protected information, we propose encryption of the logs under a homomorphic cryptosystem. The protocol requires both additive and multiplicative homomorphism. Different from the state-of-the-art, we propose an encryption switching mechanism with a partially homomorphic cryptosystem which enables to utilize both additive and multiplicative homomorphism by eliminating the necessity to use expensive alternatives. For this purpose, we select ElGamal cryptosystem [54] that is

originally multiplicatively homomorphic, but with a modification in encryption can be converted to an additive homomorphic scheme. As a result, our protocol outperforms the existing works in terms of computation with a similar level of communication cost. While providing efficiency, our protocol guarantees the protection of sensitive information in the logs. We can summarize our contributions as follows:

- We present PriSM protocol which aims to mine direct sequential patterns for business process analysis on outsourced data. Our protocol guarantees privacy protection for data owners in accordance with GDPR.
- Our proposal is based on encryption switching to avoid expensive alternatives. To the best of our knowledge, our protocol is the first work which uses both additive and multiplicative variant of ElGamal cryptosystem with a switching phase. Our protocol outperforms the existing works by using encryption switching mechanism which is at least 80% more efficient than the state-of-the-art in terms of computation cost.
- We present a security analysis for our protocol through formal security proofs that is not provided in the existing works.

In the rest of the paper, we first explain the building blocks of our protocol in Section 3.2.1. In Section 3.2.2, we explain our protocol in detail. In Section 3.2.3, we provide the analyses of our protocol with respect to security and complexity, and then we present the results of our experiments. Finally, we conclude the paper in Section 3.2.4.

3.2.1. BUILDING BLOCKS

In this section, we introduce the building blocks of our protocol. Before that we summarize the notation used throughout the paper in Table 3.6.

Table 3.6: Summary of the notation.

Symbol	Explanation
A	Set of activities s.t. $A = \{a_1, a_2, \cdots, a_\Delta\}$.
σ_i	A case with ω_i events, s.t. $\sigma_i = \langle e_{i,1}, \cdots, e_{i,\omega_i} \rangle$.
$e_{i,j}$	An event in σ_i where $1 \leq j \leq \omega_i$ and $1 \leq i \leq \tau$.
L	An event log with τ cases, s.t. $L = \{\sigma_1, \sigma_2, \cdots, \sigma_\tau\}$.
$x >_L y$	Directly-follows relation between activities x and y.
$G(L)$	Directly-follows graph of event log L.
$M_{x \times y}$	A matrix of x rows and y columns, where the size is $x \times y$.
$M_{x,y}$	Index of matrix M in row x and column y.
$M_{:,y}$	A column matrix for the y^{th} column of M.
$M_{x,:}$	A row matrix for the x^{th} row of M.
$[\cdot]^\times$	An encryption with the multiplicative variant of ElGamal.
$[\cdot]^+$	An encryption with the additive variant of ElGamal.

MINING EVENT LOGS

An event log L is a collection of actions of a system that contains information about a business process P. L is a set of cases σ_i where every event $e_{i,j} \in \sigma_i$ is unique in L, i.e., no two cases can contain the same $e_{i,j}$ [6]. Each σ_i is an instance of the process P. An event $e_{i,j}$ is an instance of activity performed by the system at any time. It can have several attributes such as the name of activity, timestamp, or the resource who performed the activity. Table 3.7 illustrates an example event log $\mathcal{L} \in L$ which contains 6 cases and 23 events. The event attributes in the example log are activity name, timestamp, and resource.

Table 3.7: An example event log, $\mathcal{L}$.

Case id	Event id	Event Attributes			
		Activity	Timestamp	Resource	⋯
σ_1	$e_{1,1}$	a	30-06-2016:11.02	Rose	⋯
	$e_{1,2}$	c	30-06-2016:13.47	Rose	⋯
	$e_{1,3}$	b	30-06-2016:16.20	Mike	⋯
	$e_{1,4}$	e	02-07-2016:10.31	Carol	⋯
σ_2	$e_{2,1}$	a	30-12-2015:08.25	Rose	⋯
	$e_{2,2}$	b	31-12-2015:11.10	Carol	⋯
	$e_{2,3}$	c	01-01-2016:09.50	Bob	⋯
	$e_{2,4}$	d	01-01-2016:09.52	Mike	⋯
σ_3	$e_{3,1}$	a	01-12-2015:13.15	Alice	⋯
	$e_{3,2}$	b	01-12-2015:14.00	Rose	⋯
	$e_{3,3}$	c	01-12-2015:14.04	Mike	⋯
	$e_{3,4}$	d	02-12-2015:10.34	Mike	⋯
σ_4	$e_{4,1}$	a	27-03-2016:11.15	Rose	⋯
	$e_{4,2}$	e	27-03-2016:11.45	Mike	⋯
	$e_{4,3}$	d	27-03-2016:13.00	Carol	⋯
σ_5	$e_{5,1}$	a	31-03-2016:10.50	Alice	⋯
	$e_{5,2}$	b	31-03-2016:14.00	Bob	⋯
	$e_{5,3}$	c	31-03-2016:14.29	Carol	⋯
	$e_{5,4}$	d	31-03-2016:17.10	Carol	⋯
σ_6	$e_{6,1}$	a	06-02-2016:16.00	Rose	⋯
	$e_{6,2}$	c	06-02-2016:16.20	Mike	⋯
	$e_{6,3}$	b	06-02-2016:17.06	Bob	⋯
	$e_{6,4}$	e	06-02-2016:21.19	Carol	⋯

A behaviour graph of a process can be generated from any attribute depending on the aim of behaviour analysis. A common practice is to generate the graph from activity names and then extend it with additional attributes such as time or resource. Then, L can be simplified as follows:

Definition 3.2.1. A simple event log L is a multi-set of cases, where every σ_i is formed by a set of activities, i.e., $e_{i,j} = a_k$, such that [6]:

$$L = \left\{ \langle e_{1,1}, \cdots, e_{1,\omega_1} \rangle^{z_1}, \cdots, \langle e_{\tau,1}, \cdots, e_{\tau,\omega_\tau} \rangle^{z_\tau} \right\}, \tag{3.1}$$

where z_i is the frequency of σ_i. Regarding Definition 3.2.1, we can formulate the log in Table 3.7 as a simple event log:

$$\mathcal{L} = \left\{\langle a,c,b,e\rangle^2, \langle a,b,c,d\rangle^3, \langle a,e,d\rangle\right\}. \tag{3.2}$$

Discovering a behaviour graph for a business process P requires to mine direct sequential patterns, a.k.a directly-follows relations, in L. An activity pair (x, y) has directly-follows relation, $x >_L y$, if x is directly followed by y in σ_i. The set of the directly-follows relations of L can be represented as a directly-follows graph, $G(L)$, which represents the behaviour of P. $G(L)$ consists of

- A_L, the set of activities in L,
- $>_L$, the set of directly-follows relations in L,
- ${A_L}^{\text{start}}$, the set of start activities in L,
- ${A_L}^{\text{end}}$, the set of end activities in L [6].

The directly-follows graph of $\mathcal{L}$, $G(\mathcal{L})$, is represented as $G(\mathcal{L}) = (A_\mathcal{L}, >_\mathcal{L}, {A_\mathcal{L}}^{\text{start}}, {A_\mathcal{L}}^{\text{end}})$ such that

$$A_\mathcal{L} = \{a,b,c,d,e\},\ A_\mathcal{L}^{\text{start}} = \left\{\{a\}^6\right\},\ A_\mathcal{L}^{\text{end}} = \left\{\{d\}^4, \{e\}^2\right\},$$
$$>_\mathcal{L} = \left\{(a,b)^3, (a,c)^2, (a,e)^1, (b,c)^3, (c,b)^2, (b,e)^2, (c,d)^3, (e,d)^1\right\}.$$

Figure 3.4 illustrates $G(\mathcal{L})$. The activities are represented by square nodes and directly-follows relations between activity pairs are represented as arrows. The numbers on arrows show the frequency of directly-follows relation. The dashed arrows belong to start and end activities.

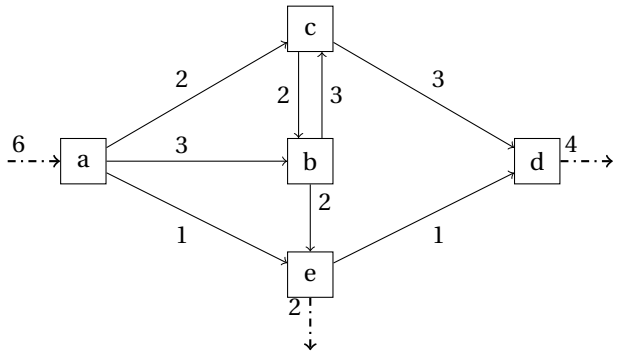


Figure 3.4: Illustration of the directly-follows graph $G(\mathcal{L})$.

In this paper, our aim is to design a protocol which mines the direct sequential patterns from an event log L and combines them to generate directly-follows graph $G(L)$.

ELGAMAL ENCRYPTION

Our protocol relies on ElGamal cryptosystem [54]. The security of ElGamal cryptosystem depends on the decisional Diffie-Hellman assumption. In the setup of the encryption scheme, three parameters p, q, and g are selected such that p and q are two large primes, where $q|(p-1)$, and g is a generator of a group $\mathcal{G}_q$, where q is the order of $\mathcal{G}$ in modulus p. For key generation, the private key $s \in_R Z_q$ is selected randomly and the corresponding public key $h = g^s$ is computed. To encrypt a message $m \in \mathcal{G}_q$ one selects $r \in_R Z_q$ and computes the ciphertext c as $c = (x, y) = (g^r, m \cdot h^r)$. For decryption of c, one computes $\frac{y}{x^s} = \frac{m \cdot g^{rs}}{g^{rs}}$ and retrieves the message m.

In this setting, ElGamal encryption satisfies partial homomorphism on multiplication operation. Given encryption of two messages m_1, m_2 as $c_1 = (x_1, y_1) = (g^{r_1}, m_1 \cdot h^{r_1})$ and $c_2 = (x_2, y_2) = (g^{r_2}, m_2 \cdot h^{r_2})$, the inner product of two messages $(x_1 \cdot x_2,\ y_1 \cdot y_2)$ results in their multiplication:

$$(x_1 \cdot x_2,\ y_1 \cdot y_2) = (g^{r_1+r_2}, (m_1 \cdot m_2) \cdot h^{r_1+r_2}). \tag{3.3}$$

Additive homomorphism can be also achieved in ElGamal cryptosystem by modifying the encryption function as $c = (g^r, g^m \cdot h^r)$, where $m \in Z_q$ [55]. Since $c_1 = (x_1, y_1) = (g^{r_1}, g^{m_1} \cdot h^{r_1})$ and $c_2 = (x_2, y_2) = (g^{r_2}, g^{m_2} \cdot h^{r_2})$, addition under encryption is performed as

$$(x_1 \cdot x_2,\ y_1 \cdot y_2) = (g^{r_1+r_2}, g^{(m_1+m_2)} \cdot h^{r_1+r_2}). \tag{3.4}$$

The shortcoming of the additive homomorphic variant is that the decryption results in $g^{m_1+m_2}$, which incurs a discrete logarithm problem. Decryption can be performed with brute-force mechanism but it limits the message space. Thus, additive homomorphism in ElGamal is feasible only for small message spaces.

A common problem in the practicability of ElGamal cryptosystem is message encoding. Since the message space in ElGamal is $\mathcal{G}_q$, the plaintext values should be mapped to the group elements before encryption. To overcome the message encoding problem, we use $m \rightarrow g^m$ as encoding function. This encoding has a shortcoming due to limited message space. Since our input size is significantly shorter compared to parameters of ElGamal encryption, such an encoding is feasible for our scheme. To clarify the notation, for the multiplicative variant of ElGamal m is first encoded to g^m and the encryption is represented as $[m]^{\times}$. For the additive variant of the cryptosystem, m does not change but encryption function is computed as g^m and the encryption of additive variant is represented as $[m]^{+}$.

3.2.2. PRISM: PRIVACY-PRESERVING SEQUENTIAL PATTERN MINING

In this section, we describe our Privacy-preserving Sequential pattern Mining protocol, PriSM. Our scenario consists of two parties which are a data owner (DO) and a data analyst (DA). DA runs PriSM protocol under the supervision of DO to mine all direct sequential patterns in L. The goal of the PriSM protocol is to generate an encrypted directly-follows graph $G(L)^+$ from encrypted L. We assume that the input L is a simplified event log as explained in Section 3.2.1 following the common practice in behaviour model discovery. Besides, we are aware of the fact that there can be other information in event logs

which are privacy sensitive. Since our protocol is based on the simplified event log, all other information stored in L is not within the scope of our paper. We assume that DA is not allowed to access any information in L apart from the activity names, which are shared with DA only under encryption.

Our setting assumes a semi-honest security model, where DO and DA do not collude. In this security model, parties exactly follow the requirements of a protocol, but they might try to obtain some information by observing the intermediary messages they receive. It is a sufficient security model when the parties trust each other in following the requirements of the protocol but they do not want to leak any inadvertent information [56]. Semi-honest security is an appropriate assumption in our setting, since both DO and DA are corporations that avoid malicious acts considering their reputation and legal restrictions. In this setting, the adversarial power is limited to computationally bounded, nonadaptive adversaries. In Section 3.2.3 we discuss the security of PriSM protocol with respect to these assumptions.

PRISM PROTOCOL

In the setup phase of PriSM protocol, DO generates the key pair for ElGamal as (s,h) and shares h with DA together with other public parameters p,q, and g. Every event $e_{i,j}$ is encrypted with the multiplicative ElGamal as $[e_{i,j}]^{\times} = (g^r, g^{e_{i,j}})$ before it is included in L. The encrypted form of L becomes

$$[L]^{\times} = \left[\langle [e_{1,1}]^{\times}, \ldots, [e_{1,\omega_1}]^{\times}\rangle, \ldots, \langle [e_{\tau,1}]^{\times}, \ldots, [e_{\tau,\omega_\tau}]^{\times}\rangle\right]. \tag{3.5}$$

Similarly, DO encrypts activities as $[A]^{\times} = \left\{[a_1]^{\times}, \ldots [a_\Delta]^{\times}\right\}$ and sends $[L]^{\times}$ and $[A]^{\times}$ to DA. Using $[L]^{\times}$ and $[A]^{\times}$ as inputs, DA runs PriSM protocol as explained in Algorithm 4. Regarding the homomorphic property of ElGamal, the protocol consists of three phases which are *multiplication phase*, *switching phase*, and *addition phase*. In the multiplication phase every $[\sigma_i]^{\times}$ in $[L]^{\times}$ is analyzed to identify the activities and direct sequential patterns secretly. In the switching phase the multiplicative variant of ElGamal cryptosystem is converted to the additive variant. In the addition phase the results are aggregated into the output $[G(L)]^{+}$.

Algorithm 4 PriSM

Input: $[L]^{\times}$, $[A]^{\times}$

for all $[\sigma_i]^{\times} \in [L]^{\times}$ **do**
  */*Multiplication phase*/*
  $[I_{\Delta\times\omega_i}]^{\times} \leftarrow$ Identification$([\sigma_i]^{\times}, [A]^{\times})$
  for all $[I_{:,j}]^{\times} \in [I_{\Delta\times\omega_i}]^{\times}$ **do**
    $[S_{\Delta\times\Delta}]^{\times} \leftarrow$ Succession$([I_{:,j}]^{\times})$
    */*Switching phase*/*
    $[S_{\Delta\times\Delta}]^{+} \leftarrow$ SwitchEncryption$([S_{\Delta\times\Delta}]^{\times})$
    */*Addition phase*/*
    $[G(L)]^{+} \leftarrow$ Aggregation$([S_{\Delta\times\Delta}]^{+})$

Output: $[G(L)]^{+}$

Multiplication Phase In this phase, first the distinct activities (A_L, A_L^{start}, A_L^{end}) in each $[\sigma_i]^\times$ are identified by an Identification subprotocol. The result of this subprotocol is used to discover direct succession relations ($>_L$) for every $[\sigma_i]^\times$ in a Succession subprotocol.

Identification: The first task in PriSM is to discover A_L, A_L^{start}, and A_L^{end} from $[L]^\times$ in a privacy-preserving manner. The naive approach to discover these sets under encryption is to compare every $[a_k]^\times \in [A]^\times$ to every $[e_{i,j}]^\times \in [L]^\times$ using a secure equality check protocol. Performing such a protocol is expensive in terms of computation and communication, and its repetition in large numbers creates a significant bottleneck in the performance [52]. In the Identification subprotocol (Algorithm 5), we overcome this bottleneck by performing arithmetic operations which enable us to secretly identify the activities. The subprotocol takes $[\sigma_i]^\times$ and $[A]^\times$ as input and multiplies every $[e_{i,j}]^\times \in [L]^\times$ with the multiplicative inverse of each $[a_k]^\times \in [A]^\times$. The result of the multiplication, stored in $[I_{\Delta\times\omega_i}]^\times$, becomes $[1]^\times$ if and only if the activity name of $[e_{i,j}]^\times$ is equal to $[a_k]^\times$. If an index in row matrix $[I_{k,:}]^\times$ is equal to $[1]^\times$, then an activity $[a_k]^\times$ is observed in $[\sigma_i]^\times$. The inverse values for $[A]^\times$ can be computed in the setup phase.

Algorithm 5 Identification

Input: $[\sigma_i]^\times$, $[A]^\times$

1: **for all** $[a_k]^\times \in [A]^\times$ **do**
2: **for all** $[e_{i,j}]^\times \in [\sigma_i]^\times$ **do**
3: $[I_{k,j}]^\times = [a_k^{-1}]^\times \times [e_{i,j}]^\times$

Output: $[I_{\Delta\times\omega_i}]^\times$

Figure 3.5 shows the result of Identification subprotocol which we run on σ_1 of $\mathcal{L}$ as matrix $I_{5\times4}$. For the sake of simplicity, we encode the letters into integers with respect to their alphabetical order such as $a = 1, b = 2, c = 3$. Since we use $m \to g^m$ as message encoding function in ElGamal, the original plaintext values are different than the ones illustrated in Figure 3.5. It is important to note that in the illustrations we represent the operations with simple integer arithmetic instead of the real encryption values.

$[I]^\times$	$[a]^\times$	$[c]^\times$	$[b]^\times$	$[e]^\times$
$[a^{-1}]^\times$	$[1]^\times$	$[3]^\times$	$[2]^\times$	$[5]^\times$
$[b^{-1}]^\times$	$\left[\frac{1}{2}\right]^\times$	$\left[\frac{3}{2}\right]^\times$	$[1]^\times$	$\left[\frac{5}{2}\right]^\times$
$[c^{-1}]^\times$	$\left[\frac{1}{3}\right]^\times$	$[1]^\times$	$\left[\frac{2}{3}\right]^\times$	$\left[\frac{5}{3}\right]^\times$
$[d^{-1}]^\times$	$\left[\frac{1}{4}\right]^\times$	$\left[\frac{3}{4}\right]^\times$	$\left[\frac{2}{4}\right]^\times$	$\left[\frac{5}{4}\right]^\times$
$[e^{-1}]^\times$	$\left[\frac{1}{5}\right]^\times$	$\left[\frac{3}{5}\right]^\times$	$\left[\frac{2}{5}\right]^\times$	$[1]^\times$

Figure 3.5: Performing Identification subprotocol on $\sigma_1 \in \mathcal{L}$.

Succession: The second task in PriSM protocol is to find direct sequential patterns between activities. Since every column of $[I]^\times$ matrix carries information about one iden-

tified activity, correlation of two subsequent columns with a function can give us the directly-follows relation between two activities. One way to do that is to multiply two columns such that in the output matrix only the index corresponding to the discovered activity pair becomes 1 and the rest of indices are dummy values. As demonstrated in Algorithm 6, we multiply every j^{th} column of $[I]^{\times}$ with the transpose of $(j+1)^{\text{th}}$ column. To prevent false positives in the multiplication, we apply an interpolation which shifts every value in $[I_{:,j}]^{\times}$ by $2\Delta+1$. We select $2\Delta+1$ as the shifting factor to avoid collusions. Since the value for every index of $[I]^{\times}$ is either in $[0,\Delta)$ or $[q-\Delta,q)$, shifting the operations to $[\Delta,q-\Delta)$, which is achieved by $2\Delta+1$, can prevent false positives. Since $q \gg \Delta$, we guarantee that $[S]^{\times}$ contains a single $[1]^{\times}$.

Algorithm 6 Succession

Input: $[I_{\Delta\times\omega_i}]^{\times}$

1: **for** $1 \le j \le \omega_i - 1$ **do**

2: $\quad [S_{\Delta\times\Delta}] \leftarrow ([I_{:,j}]^{\times})^{2\Delta+1} \times ([I_{:,j+1}]^{\times})^{T}$

Output: $[S_{\Delta\times\Delta}]^{\times}$

In Figure 3.6, we apply Succession subprotocol on the first two columns of $[I]^{\times}$. Notice that in Figure 3.5 activities a and c are secretly discovered in the first two columns. By assigning $\Delta = 5$ for the interpolation, in $[S]^{\times}$ only the index (1,3), which corresponds to (a,c) pair, becomes $[1]^{\times}$ and the rest becomes dummy.

$[S]^{\times}$	$[3]^{\times}$	$\left[\frac{3}{2}\right]^{\times}$	$[1]^{\times}$	$\left[\frac{3}{4}\right]^{\times}$	$\left[\frac{3}{5}\right]^{\times}$
$[1^{11}]^{\times}$	$[3]^{\times}$	$\left[\frac{3}{2}\right]^{\times}$	$[1]^{\times}$	$\left[\frac{3}{4}\right]^{\times}$	$\left[\frac{3}{5}\right]^{\times}$
$\left[\frac{1}{2}^{11}\right]^{\times}$	$\left[\frac{3}{2^{11}}\right]^{\times}$	$\left[\frac{3}{2^{12}}\right]^{\times}$	$\left[\frac{1}{2^{11}}\right]^{\times}$	$\left[\frac{3}{2^{13}}\right]^{\times}$	$\left[\frac{3}{5\cdot 2^{11}}\right]^{\times}$
$\left[\frac{1}{3}^{11}\right]^{\times}$	$\left[\frac{1}{3^{10}}\right]^{\times}$	$\left[\frac{1}{2\cdot 3^{10}}\right]^{\times}$	$\left[\frac{1}{3^{11}}\right]^{\times}$	$\left[\frac{1}{4\cdot 3^{10}}\right]^{\times}$	$\left[\frac{1}{5\cdot 3^{10}}\right]^{\times}$
$\left[\frac{1}{4}^{11}\right]^{\times}$	$\left[\frac{3}{4^{11}}\right]^{\times}$	$\left[\frac{3}{2\cdot 4^{11}}\right]^{\times}$	$\left[\frac{1}{4^{11}}\right]^{\times}$	$\left[\frac{3}{4^{12}}\right]^{\times}$	$\left[\frac{3}{5\cdot 4^{11}}\right]^{\times}$
$\left[\frac{1}{5}^{11}\right]^{\times}$	$\left[\frac{3}{5^{11}}\right]^{\times}$	$\left[\frac{3}{2\cdot 5^{11}}\right]^{\times}$	$\left[\frac{1}{5^{11}}\right]^{\times}$	$\left[\frac{3}{4\cdot 5^{11}}\right]^{\times}$	$\left[\frac{3}{5^{12}}\right]^{\times}$

Figure 3.6: Performing Succession subprotocol on the two columns of $[I]^{\times}$.

Switching Phase During the multiplication phase, we discover the set of activities and the set of directly-follows relations using homomorphic properties of encryption. The final task for creating a directly-follows graph is to calculate the frequency of directly-follows relations, the start, and end activities. Since in the current state ElGamal is in multiplicative variant, performing additions to calculate frequencies is not trivial. Thus, we introduce an interactive switching phase which transforms the cryptosystem from multiplicative variant to the additive variant through a secure decryption.

As demonstrated in Algorithm 7, DA blinds every index of $[S]^{\times}$ with a fresh $r_{i,j} \in_R Z_q$ and sends the blinded $[S^r]^{\times}$ to DO for secure decryption. Before sharing $[S^r]^{\times}$ with DO,

Algorithm 7 SwitchEncryption

DA $([S]^{\times})$		**DO** (s)
1) blind:		
$[S^r]^{\times} \Leftarrow [S]^{\times}$ such that		
$[S_{i,j}^{r_{i,j}}]^{\times} \leftarrow ([S_{i,j}]^{\times})^{r_{i,j}}, r_{i,j} \in_R Z_q$		
2) shuffle:		
$[S_p^r]^{\times} \Leftarrow p([S]^{\times})$	$\xrightarrow{[S_p^r]^{\times}}$	
		3) decrypt:
		$S_p^r \Leftarrow \mathrm{Dec}([S_p^r]^{\times}, s)$
		4) map:
		if $S_{p_{i,j}}^r = 1 \rightarrow [S_{p_{i,j}}]^+ = [1]^+$
	$\xleftarrow{[S_p]^+}$	**else** $\rightarrow [S_{p_{i,j}}]^+ = [0]^+$
5) revert shuffling:		
$[S]^+ \Leftarrow p^{(-1)}([S_p]^+)$		

DA shuffles $[S^r]^{\times}$ with a shuffling function $p(\cdot)$ where the shuffled matrix is $[S_p^r]^{\times}$. The decryption works as a mapping function. If the result of decryption is 1, then DO encrypts 1 in the additive variant, i.e., $[1]^+$, else he encrypts 0 as $[0]^+$. The result of decryption is 1 only for the index which contains the direct succession relation as blinding does not change its value. The result is random for the other indices so decryption does not leak any information about them. Since by default $[S]^{\times}$ contains a single $[1]^{\times}$, revealing it to DA does not degrade the security as soon as its location is hidden by shuffling. In Figure 3.7, we illustrate the result of switching phase for $[\mathcal{S}]^{\times}$ of Figure 3.6.

$[\mathcal{S}]^+$	a	b	c	d	e
a	$[0]^+$	$[0]^+$	$[1]^+$	$[0]^+$	$[0]^+$
b	$[0]^+$	$[0]^+$	$[0]^+$	$[0]^+$	$[0]^+$
c	$[0]^+$	$[0]^+$	$[0]^+$	$[0]^+$	$[0]^+$
d	$[0]^+$	$[0]^+$	$[0]^+$	$[0]^+$	$[0]^+$
e	$[0]^+$	$[0]^+$	$[0]^+$	$[0]^+$	$[0]^+$

Figure 3.7: $[\mathcal{S}]^+$ matrix as the result of encryption switching phase.

Addition Phase The final task of our protocol is to calculate the frequencies of direct sequential patterns. Since the encryption is switched to the additive variant, performing additions is now feasible. In this phase, every $[S]^+$ as the output of switching phase is aggregated into one matrix by homomorphic addition. The result is $[G(\mathcal{L})]^+$ which contains the activities, their directly-follows relations, and frequencies under encryption. In

Figure 3.8 we show the result for $\mathcal{L}$. Notice that to avoid the complication in representation, we do not include the sets $A_{\mathcal{L}}^{\text{start}}$ and $A_{\mathcal{L}}^{\text{end}}$ in the figure. However, they are discovered along with $A_{\mathcal{L}}$ by only excluding Succession subprotocol.

$[G(\mathcal{L})]^+$	a	b	c	d	e
a	$[0]^+$	$[3]^+$	$[2]^+$	$[0]^+$	$[1]^+$
b	$[0]^+$	$[0]^+$	$[3]^+$	$[0]^+$	$[2]^+$
c	$[0]^+$	$[2]^+$	$[0]^+$	$[3]^+$	$[0]^+$
d	$[0]^+$	$[0]^+$	$[0]^+$	$[0]^+$	$[0]^+$
e	$[0]^+$	$[0]^+$	$[0]^+$	$[1]^+$	$[0]^+$

Figure 3.8: $[G(\mathcal{L})]^+$ matrix as the output of the protocol.

3.2.3. PROTOCOL ANALYSES

In this section we analyze PriSM protocol for its security and complexity in terms of communication and computation, then provide the timing results of experiments.

SECURITY OF PRISM

Our protocol aims to securely process the sensitive data in event logs. In the semi-honest setting, parties DA and DO should not be able to retrieve any additional information apart from their inputs, outputs, and intermediary messages. During the non-interactive phases of our protocol, multiplication and addition, security is guaranteed by the semantic security of ElGamal cryptosystem under decisional Diffie-Hellman assumption. In the interactive switching phase, however, a security analysis is necessary to formally prove the security despite semi-honest assumption limits the adversarial power. In the rest, we prove the security of the switching phase based on simulation paradigm [56].

The goal in simulation paradigm is to show the security of a protocol by comparing the view of an adversary in the real world to the simulated view in the ideal world, where the security is guaranteed [57]. If an adversary $\mathcal{A}$ can attack the protocol in the real world, the attack can be also performed by the adversary of the ideal world, $\mathcal{S}$. Since the attacks of $\mathcal{S}$ are not successful in the ideal setting, the attacks in the real world also fail and the protocol is proved to be secure in the real world.

Computational Indistinguishability [56]: Let $X(a,\kappa)$ and $Y(a,\kappa)$ are two distribution ensembles where $a \in \{0,1\}^*$ is input and κ is the security parameter. $X(a,\kappa)$ and $Y(a,\kappa)$ are computationally indistinguishable ($X(a,\kappa) \stackrel{c}{\equiv} Y(a,\kappa)$) if there exist a negligible function $\mu(\kappa)$ for every nonuniform polynomial time algorithm D such that

$$|\Pr[D(X(a,\kappa)) = 1] - \Pr[D(Y(a,\kappa)) = 1]| \leq \mu(\kappa). \tag{3.6}$$

Definition of Security [56]: Let π be a two-party protocol between parties P_1 and P_2 to compute functionality $f(x,y)$ on the inputs x and y, where $f_1(x,y)$ and $f_2(x,y)$ represent

the result of $f(x,y)$ for the parties. During the execution of π the view of the parties are:

$$\mathbf{view}_1^{\pi}(x,y,\kappa) = (x, r_1; m_1, m_2, \cdots, m_t), \tag{3.7}$$
$$\mathbf{view}_2^{\pi}(x,y,\kappa) = (y, r_2; m_1, m_2, \cdots, m_t), \tag{3.8}$$

where r_1, r_2 are the randomness of the parties, κ is the security parameter and m_is are the intermediary messages of the parties. Let

$$\mathbf{output}^{\pi}(x,y,\kappa) = (\mathbf{output}_1^{\pi}(x,y,\kappa), \mathbf{output}_2^{\pi}(x,y,\kappa)) \tag{3.9}$$

represent the joint output of π, where $\mathbf{output}_1^{\pi}(x,y,\kappa)$ and $\mathbf{output}_2^{\pi}(x,y,\kappa)$ are the local outputs of P_1 and P_2, respectively. The protocol π securely computes $f(x,y)$ in the semi-honest setting against nonadaptive, computationally bounded adversaries, if there exist probabilistic polynomial time simulators $\mathcal{S}_1$ and $\mathcal{S}_2$ such that

$$\left\{\mathcal{S}_1(1^{\kappa}, x, f_1(x,y)), f(x,y)\right\} \overset{c}{\equiv} \left\{\mathbf{view}_1^{\pi}(x,y,\kappa), \mathbf{output}^{\pi}(x,y,\kappa)\right\}, \tag{3.10}$$
$$\left\{\mathcal{S}_2(1^{\kappa}, y, f_2(x,y)), f(x,y)\right\} \overset{c}{\equiv} \left\{\mathbf{view}_2^{\pi}(x,y,\kappa), \mathbf{output}^{\pi}(x,y,\kappa)\right\}. \tag{3.11}$$

In the switching phase, π is a protocol between DA and DO which computes a functionality f. DA's input is $[S]^{\times}$ while SC's input is an empty string, $\perp$. f computes $[S]^{+}$ from $[S]^{\times}$ such that $f([S]^{\times}, \perp) = ([S]^{+}, \perp)$, where $[S]^{+}$ and $\perp$ are the outputs of DA and DO, respectively.

Theorem 3.2.1. The encryption switching protocol π securely computes the functionality $f([S]^{\times}, \perp) = ([S]^{+}, \perp)$ in the presence of semi-honest adversaries.

Proof. To prove the security we show that the view of adversary $\mathcal{A}$ in the real world is computationally indistinguishable from the view of a simulator $\mathcal{S}_i$ where $i \in \{\text{DA, DO}\}$ in the ideal world.

DA is corrupted by $\mathcal{A}$: Given that $\mathcal{S}_{\text{DA}}$ has access to the input and output of DA, $\mathcal{S}_{\text{DA}}$ simulates the view of the messages DA receives as follows:

1. $\mathcal{S}_{\text{DA}}$ choses a uniformly distributed random, r_1.
2. $\mathcal{S}_{\text{DA}}$ creates a matrix S^* of size $\Delta \times \Delta$ which contains a single 1 and $(\Delta^2 - 1)$ 0s where the location of 1 is chosen randomly.
3. $\mathcal{S}_{\text{DA}}$ shuffles S^*, such that $S_p^* \leftarrow p(S^*; r_1)$.
4. $\mathcal{S}_{\text{DA}}$ encrypts S_p^* under additive variant of ElGamal cryptosystem as $[S_p^*]^{+}$.

The simulated view of $\mathcal{S}_{\text{DA}}$ is

$$\mathcal{S}_{\text{DA}}(1^{\kappa}, [S]^{\times}, [S]^{+}) = \left([S]^{\times}, r_1; [S_p^*]^{+}\right), \tag{3.12}$$

and the output is $f([S]^{\times}, \perp) = ([S]^{+}, \perp)$. On the other hand, the view of $\mathcal{A}$ is

$$\mathbf{view}_{\text{DA}}^{\pi}([S]^{\times}, \perp) = \left([S]^{\times}, r_{\text{DA}}; [S_p]^{+}\right), \tag{3.13}$$

where $\mathbf{output}^{\pi}([S]^{\times},\perp)=\left([S]^{+},\perp\right)$. Then,

$$\left\{\mathcal{S}_{DA}(1^{\kappa},[S]^{\times},[S]^{+}),f([S]^{\times},\perp)\right\}\stackrel{c}{\equiv}\left\{\mathbf{view}^{\pi}_{\mathrm{DA}}([S]^{\times},\perp)\mathbf{output}^{\pi}([S]^{\times},\perp)\right\},\tag{3.14}$$

if for every nonuniform polynomial time distinguisher D there exists a negligible function $\mu(\kappa)$ such that

$$\left|\Pr\left[D\left(\left([S]^{\times},r_1;[S^{*}_{p}]^{+}\right)\wedge\left([S]^{+},\perp\right)\right)=1\right]-\Pr\left[D\left(\left([S]^{\times},r_{\mathrm{DA}};[S_p]^{+}\right)\wedge\left([S]^{+},\perp\right)\right)=1\right]\right|\leq\mu(\kappa).\tag{3.15}$$

Equation 3.15 holds if a semantically secure cryptosystem is chosen for the encryption which is ElGamal in our protocol. The indistinguishability guarantees that $\mathcal{S}_{\mathrm{DA}}$ gains no advantage on correctly guessing the location of $[1]^{\times}$ in $[S]^{\times}$. The probability of $[1]^{\times}$ located in $[S_{i,j}]^{\times}$ is equal to $\frac{1}{\Delta^2}$. Similarly, when creating S^{*}, the probability of $\mathcal{S}_{\mathrm{DA}}$ selecting (i,j) as the index of 1 is $\frac{1}{\Delta^2}$.

DO is corrupted by $\mathcal{A}$: Different from the former case, DO does not have an input and output, thus, $\mathcal{S}_{\mathrm{DO}}$ can only receive an auxiliary input which is the private key of DO, s. With the provided information $\mathcal{S}_{\mathrm{DO}}$ behaves as follows:

1. $\mathcal{S}_{\mathrm{DO}}$ choses uniformly distributed randoms, r_2,r_3.
2. $\mathcal{S}_{\mathrm{DO}}$ creates a matrix S^{*} of size $\Delta\times\Delta$ which contains a single 1 and (Δ^2-1) 0s where the location of 1 is chosen randomly.
3. $\mathcal{S}_{\mathrm{DO}}$ encrypts S^{*} under the multiplicative variant of the ElGamal cryptosystem as $[S^{*}]^{\times}$.
4. $\mathcal{S}_{\mathrm{DO}}$ blinds every index of $[S^{*}]^{\times}$ with a fresh random $r_{i,j}$ which is uniformly sampled from r_2 and obtains $[S^{r*}]^{\times}$.
5. $\mathcal{S}_{\mathrm{DO}}$ shuffles $[S^{r*}]^{\times}$ such that $[S^{r}_{p}{}^{*}]^{\times}\leftarrow p([S^{r*}]^{\times};r_3)$.

The simulated view of $\mathcal{S}_{\mathrm{DO}}$ is

$$\mathcal{S}_{DO}(1^{\kappa},\perp,\perp)=\left(\perp,r_2,r_3;[S^{r}_{p}{}^{*}]^{\times}\right)\tag{3.16}$$

and the view of $\mathcal{A}$ is $\mathbf{view}^{\pi}_{\mathrm{DO}}([S]^{\times},\perp)=\left(\perp,r_{\mathrm{DO}};[S^{r}_{p}]^{\times}\right)$. Since the security definition requires the indistinguishability of joint outputs, the output of $f([S]^{\times},\perp)$ in $\mathcal{S}_{DO}$'s view and the output of π in DO's view are the same with the former case, i.e., $f([S]^{\times},\perp)=([S]^{+},\perp)$ and $\mathbf{output}^{\pi}([S]^{\times},\perp)=\left([S]^{+},\perp\right)$. We can prove the indistinguishability as follows:

$$\left\{\mathcal{S}_{DO}(1^{\kappa},\perp,\perp),f([S]^{\times},\perp)\right\}\stackrel{c}{\equiv}\left\{\mathbf{view}^{\pi}_{\mathrm{DO}}([S]^{\times},\perp),\mathbf{output}^{\pi}([S]^{\times},\perp)\right\}\tag{3.17}$$

if for every nonuniform polynomial time distinguisher D there exists a negligible function $\mu(\kappa)$ such that

$$\left|\Pr\left[D\left(\left(\perp,r_2,r_3;[S^{r}_{p}{}^{*}]^{\times}\right)\wedge\left([S]^{+},\perp\right)\right)=1\right]-\Pr\left[D\left(\left(\perp,r_{\mathrm{DO}};[S^{r}_{p}]^{\times}\right)\wedge\left([S]^{+},\perp\right)\right)=1\right]\right|\leq\mu(\kappa).\tag{3.18}$$

The indistinguishability guarantees that $\mathcal{S}_{\text{DO}}$ has no advantage on correctly guessing the real location of 1 in $[S_p^r]^\times$. The probability of 1 located in $[S_{i,j}^{r_p}]^\times$ is equal to $\frac{1}{\Delta^2}$. Similarly, when creating $[S_p^{r*}]^\times$, the probability of $\mathcal{S}_{\text{DO}}$ selecting (i, j) as the index of 1 is $\frac{1}{\Delta^2}$. Due to the semantic security of ElGamal, observing $[S_p^r]^\times$ does not give $\mathcal{S}_{\text{DO}}$ any additional advantage on correctly guessing the location. □

COMPLEXITY ANALYSIS

In the following subsections we present a complexity analysis for PriSM protocol to evaluate its feasibility. Furthermore, we compare our protocol with the AlphaSec protocol in [52] to analyze the efficiency of both protocols. We could not provide a comparison with [53] since no analysis is shown apart from the cost of encryptions and decryptions. To clarify the comparison, we use a similar notation with [52], which is summarized in Table 3.8.

Table 3.8: notation used for complexity analyses.

Symbol	Explanation
Δ	The number of elements in activity set A.
p	The modulus for ElGamal encryption.
ω_i	The number of elements in σ_i.
τ	Total number of cases (σ_i's) in L.
Γ	Total number of events in L s.t. $\Gamma = \sum_{i=1}^{\tau} \omega_i$.

Communication Complexity During encryption switching phase our protocol requires an interactive protocol between DO and DA. For one encryption switching, DA sends $[S]^\times$ matrix whose size is $2 \cdot \Delta^2 \cdot |p|$ bits where $|p|$ is the size of encryption modulus. In reply, DO sends $[S]^+$ matrix whose size is also $2 \cdot \Delta^2 \cdot |p|$ bits. For one execution of switching phase the bandwidth usage is $4 \cdot \Delta^2 \cdot |p|$ bits. The total bandwidth usage of the protocol becomes $4 \cdot \Delta^2 \cdot (\omega_i - 1) \cdot \tau \cdot |p|$, which means the communication cost is dominated by the number of activities, i.e., Δ.

The dominant factor in the bandwidth usage of AlphaSec protocol [52] is also Δ. However, with the help of data packing, the communication cost of AlphaSec protocol can be reduced significantly. To compare how PriSM performs compared to packed (w/ Pack) and non-packed (w/o Pack) versions of AlphaSec protocol we perform a communication cost analysis for different log sizes as demonstrated in Figure 3.9. To measure the performance of AlphaSec we use the formulas provided in Table 5 of the corresponding paper [52]. We perform the analysis on 3 different datasets where $\Delta = 200$ activities for all of them and $\Gamma = 10\,000$, $\Gamma = 100\,000$, and $\Gamma = 1\,000\,000$, respectively.

The results show that our protocol surpasses non-packed AlphaSec algorithm but it cannot perform better than the packed version. The reason is that data packing reduces the complexity of AlphaSec protocol almost to linear degrees in practice while it is quadratic in theoretic bounds. Although our protocol infers less interaction, only for encryption switching, the amount of exchanged data is more than the packed version of AlphaSec.

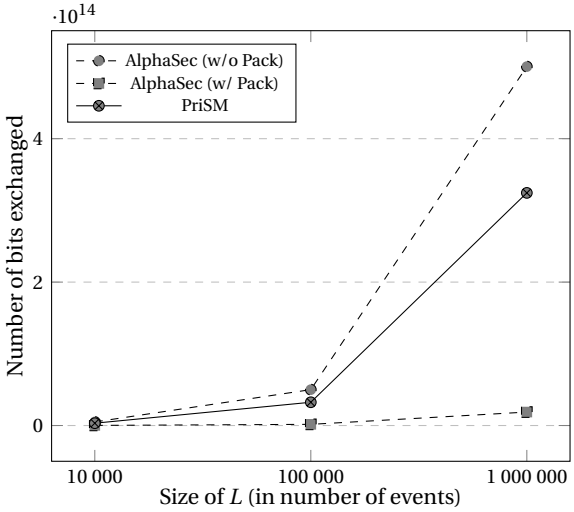


Figure 3.9: Comparison of bandwidth usage for AlphaSec and PriSM.

Table 3.9: Analysis of computational complexity for PriSM in terms of the number of operations.

	Operation				
Phase	**Encryption**	**Decryption**	**Mult.**	**Exponent.**	**ModInv.**
Setup	$\Gamma+\Delta$	–	–	–	Δ
Mult.-Ident.	–	–	$\Gamma\cdot\Delta$	–	–
Mult.-Succ.	–	–	$\Delta^2\cdot(\omega_i-1)\cdot\tau$	$\Delta\cdot(\omega_i-1)\cdot\tau$	–
SwitchEnc	$\Delta^2\cdot(\omega_i-1)\cdot\tau$	$\Delta^2\cdot(\omega_i-1)\cdot\tau$	–	$\Delta^2\cdot(\omega_i-1)\cdot\tau$	–
Aggregation	–	–	$\Delta^2\cdot(\omega_i-1)\cdot\tau$	–	–

Computational Complexity Our protocol achieves sequential pattern mining through homomorphic properties of underlying cryptosystem. In Table 3.9 we provide an analysis for the computational complexity of our protocol. The complexity of the protocol is quadratic with respect to Δ. Computing modular inverse can be performed in plaintext which incurs less computational cost. Selecting a smaller exponentiation size while setting ElGamal parameters such that $p = \alpha \cdot q + 1$ where 224 bits q and 2 048 bits p, according to current security standards [58], reduces the cost of exponentiations. Although multiplication is the most frequent operation, it is cheaper compared to exponentiation. Thus, performing encryptions and decryptions during the switching phase becomes most costly operations.

In [52], the number of operations are demonstrated with respect to the number of two-party protocols. Although the number of invocations for the protocols in AlphaSec is less than PriSM, since each cryptographic two-party protocol in AlphaSec requires several encryptions and decryptions, AlphaSec becomes less practical compared to PriSM.

EXPERIMENTAL EVALUATION

We implemented PriSM protocol to evaluate its real time performance and to compare it with the state-of-the-art. We run two sets of experiments such that in the first one we compare the performance with existing work and in the second one we run it on a larger real dataset using a parallel version of our protocol. We choose C++ programming language for the implementation. For big integer operations we use GNU Multiple Precision Arithmetic library (GMP-6.1.2). We use two different machines to run our experiments. First machine runs an OSX El Capitan operating system with an Intel Core i5 2.7 GHz processor. To run experiments on larger datasets with a parallel implementation we use a second machine which runs 64-bit CentOS 7.3.1611 on 8 cores where each core is an Intel Xeon E5345 clocked at 2.33 GHz. We use OpenMP to perform parallel operations. To meet current security standards [58], we choose p as 2048 bits and q as 224-bits for ElGamal cryptosystem.

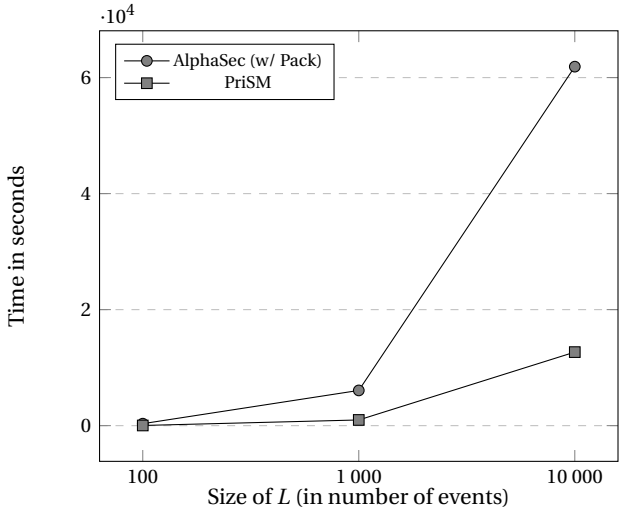


Figure 3.10: Comparison of computation cost for AlphaSec and PriSM.

In the first experiments, we compare the performance of PriSM protocol to AlphaSec protocol. We generate 3 synthetic datasets with the same properties as in [52] such that $\Gamma = 109$, $\Gamma = 1\,226$, and $\Gamma = 10\,696$ events. Figure 3.10 demonstrates the results of comparison. As the size of data increases the difference between the computational performance of AlphaSec and PriSM becomes more clear. For a dataset of 10 696 events while AlphaSec protocol requires 61 885 seconds, our protocol can be executed in 12 679 seconds which results in a 80% improvement in the computation cost.

In PriSM protocol most of the operations are performed independently and these independent operations are repeated in large numbers. Using a parallel implementation of the protocol can reduce the cost of computations by performing the independent operations simultaneously. As a second experiment we analyze the performance of PriSM protocol on larger data sizes using a parallel variant of the protocol. As dataset we selected BPI Challenge dataset [59] which contains real logs from an incident and problem

management system. The dataset contains $\Gamma = 65\,533$ events, $\tau = 7\,554$ cases, and $\Delta = 13$ activities. Average length of a case is $\omega \approx 9$ events. Considering the computational complexity of PriSM protocol, $\Delta^2 \cdot (\omega_i - 1) \cdot \tau$, we adapt two different approaches for parallel implementation. First, we apply parallelism on ω_i-level, i.e., operations on every column of I matrix (lines 4-9 in Protocol 4) are distributed on multiple cores. Second, we apply parallelism on Δ-level so the for loop between lines 4-9 in Protocol 4 runs sequentially but the intermediate operations are performed in parallel. We do not consider parallelisation on τ-level, since it requires larger memory usage in implementation. The timing results for two parallel implementations and the estimation for serial implementation are:

- ω_i-level → 20 533 seconds,
- Δ_i-level → 16 743 seconds, and
- serial (est.) → 101 094 seconds.

As expected, performing a parallel computation can reduce the computation time of PriSM protocol significantly. While the machine we use in our experiments runs only 8 cores, with the existing computation technologies it is realistic for a data analysis company to have machines with larger amount of cores. Computation time can be reduced gradually by increasing the number of cores. According to the results, Δ-level parallelism performs better than ω_i-level parallelism. The reason is that since the value of ω_i changes from 1 to 62 in the dataset, the utilization of cores is not as balanced as Δ-level parallelism where cores are always distributed on 13 activities. Thus, for any dataset, depending on the the size of Δ and ω_i, either Δ-level or ω_i-level parallelism can be selected. Moreover, with greater amount of available cores a nested parallelism can also be implemented.

3.2.4. CONCLUSION

With the rise of data mining as a service paradigm more and, more companies have adapted data analytics in their businesses. However, the paradigm has brought the privacy concerns of collected data along with itself. To overcome the privacy challenges in outsourced data analytics, we present a protocol which mines direct sequential patterns on the protected data. Specifically, our protocol models the behavior of business processes. Accordingly, we use a probabilistic cryptosystem to protect the sensitive data and, moreover, we use homomorphic properties of the underlying cryptosystem to process the protected data. However, working under encryption requires interactive computations which induce a heavy computation cost with the existing primitives. The novelty of our proposal is to use an encryption switching mechanism between multiplicative and additive homomorphic variants of encryption which optimizes the cost of computation by reducing the number of interactive steps. Our experiments show that using encryption switching, we can gain an improvement of 80% in computational performance compared to the alternatives based on two-party homomorphism based protocols. While there exist initial attempts to address the challenge of discovering direct sequential patterns from protected event logs, our proposal presents a generic solution

for the discovery of the relations that can be adapted to renowned algorithms for business process analysis, by outperforming the state-of-the-art with respect to computation cost.

REFERENCES

[1] W. M. P. van der Aalst, *Big software on the run: in vivo software analytics based on process mining (keynote)*, in *Proceedings of the 2015 International Conference on Software and System Process, ICSSP 2015, Tallinn, Estonia, August 24 - 26, 2015* (2015) pp. 1–5.

[2] V. A. Rubin, C. W. Günther, W. M. P. van der Aalst, E. Kindler, B. F. van Dongen, and W. Schäfer, *Process mining framework for software processes*, in *Software Process Dynamics and Agility, International Conference on Software Process, ICSP 2007, Minneapolis, MN, USA, May 19-20, 2007, Proceedings* (2007) pp. 169–181.

[3] J. Levenberg, *Why google stores billions of lines of code in a single repository*, Commun. ACM **59**, 78 (2016).

[4] D. Gluch, S. Cornella-Dorda, J. J. Hudak, G. A. Lewis, J. Walker, C. B. Weinstock, and D. Zubrow, *Model-Based Verification: An Engineering Practice*, Tech. Rep. CMU/SEI-2002-TR-021 (Carnegie Mellon University, PA, 2002).

[5] A. Pecchia and M. Cinque, *Log-based failure analysis of complex systems: Methodology and relevant applications*, in *Innovative Technologies for Dependable OTS-Based Critical Systems: Challenges and Achievements of the CRITICAL STEP Project* (Springer Milan, Milano, 2013) pp. 203–215.

[6] W. M. P. van der Aalst, *Process Mining - Data Science in Action, Second Edition* (Springer, 2016).

[7] G. Gousios, *The ghtorent dataset and tool suite*, in *Proceedings of the 10th Working Conference on Mining Software Repositories, MSR '13, San Francisco, CA, USA, May 18-19, 2013* (2013) pp. 233–236.

[8] G. Gousios, *The issue 32 incident _ an update*, (2016).

[9] M. Leemans and W. M. P. van der Aalst, *Process mining in software systems: Discovering real-life business transactions and process models from distributed systems*, in *18th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MoDELS 2015, Ottawa, ON, Canada, September 30 - October 2, 2015* (2015) pp. 44–53.

[10] G. Naumovich and N. D. Memon, *Preventing piracy, reverse engineering, and tampering*, IEEE Computer **36**, 64 (2003).

[11] C. Collberg, C. Thomborson, and D. Low, *A taxonomy of obfuscating transformations*, Tech. Rep. 148 (Department of Computer Science, The University of Auckland, New Zealand, 1997).

[12] C. S. Collberg and C. D. Thomborson, *Software watermarking: Models and dynamic embeddings*, in *POPL '99, Proceedings of the 26th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, San Antonio, TX, USA, January 20-22, 1999* (1999) pp. 311–324.

[13] D. Aucsmith, *Tamper resistant software: An implementation,* in *Information Hiding, First International Workshop, Cambridge, UK, May 30 - June 1, 1996, Proceedings* (1996) pp. 317–333.

[14] M. Grechanik, C. Csallner, C. Fu, and Q. Xie, *Is data privacy always good for software testing?* in *IEEE 21st International Symposium on Software Reliability Engineering, ISSRE 2010, San Jose, CA, USA, 1-4 November 2010* (2010) pp. 368–377.

[15] Lucia, D. Lo, L. Jiang, and A. Budi, *kbe-anonymity: test data anonymization for evolving programs,* in *IEEE/ACM International Conference on Automated Software Engineering, ASE'12, Essen, Germany, September 3-7, 2012* (2012) pp. 262–265.

[16] M. Castro, M. Costa, and J. Martin, *Better bug reporting with better privacy,* in *Proceedings of the 13th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2008, Seattle, WA, USA, March 1-5, 2008* (2008) pp. 319–328.

[17] P. Broadwell, M. Harren, and N. Sastry, *Scrash: A system for generating secure crash information,* in *Proceedings of the 12th USENIX Security Symposium, Washington, D.C., USA, August 4-8, 2003* (2003).

[18] W. Enck, P. Gilbert, S. Han, V. Tendulkar, B. Chun, L. P. Cox, J. Jung, P. D. McDaniel, and A. N. Sheth, *Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones,* ACM Trans. Comput. Syst. **32**, 5:1 (2014).

[19] D. Y. Zhu, J. Jung, D. Song, T. Kohno, and D. Wetherall, *Tainteraser: protecting sensitive data leaks using application-level taint tracking,* Operating Systems Review **45**, 142 (2011).

[20] W. M. P. van der Aalst, T. Weijters, and L. Maruster, *Workflow mining: Discovering process models from event logs,* IEEE Trans. Knowl. Data Eng. **16**, 1128 (2004).

[21] P. Paillier, *Public-key cryptosystems based on composite degree residuosity classes,* in *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding* (1999) pp. 223–238.

[22] Z. Erkin, T. Veugen, T. Toft, and R. L. Lagendijk, *Generating private recommendations efficiently using homomorphic encryption and data packing,* IEEE Trans. Information Forensics and Security **7**, 1053 (2012).

[23] M. Nateghizad, Z. Erkin, and R. L. Lagendijk, *Efficient and secure equality tests,* in *IEEE International Workshop on Information Forensics and Security, WIFS 2016, Abu Dhabi, United Arab Emirates, December 4-7, 2016* (2016) pp. 1–6.

[24] R. Lazzeretti, *Privacy preserving processing of biomedical signals with application to remote healthcare systems,* Ph.D. thesis, Ph. D. thesis, PhD school of the University of Siena, Information Engineering and Mathematical Science Department (2012).

[25] H. Lipmaa and T. Toft, *Secure equality and greater-than tests with sublinear online complexity,* in *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part II* (2013) pp. 645–656.

[26] A. Aspili, *Outsourcing data analytics,* (2015).

[27] Z. Tari, X. Yi, U. S. Premarathne, P. Bertók, and I. Khalil, *Security and privacy in cloud computing: Vision, trends, and challenges,* IEEE Cloud Computing **2**, 30 (2015).

[28] A. M. Research, *World data analytics outsourcing market is expected to reach $5.9 billion, by 2020,* (2016).

[29] A. Marrella, A. Monreale, B. Klöpper, and M. W. Krueger, *Privacy-preserving outsourcing of pattern mining of event-log data - A use-case from process industry,* in *2016 IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2016, Luxembourg, December 12-15, 2016* (2016) pp. 545–551.

[30] A. Monreale and W. H. Wang, *Privacy-preserving outsourcing of data mining,* in *40th IEEE Annual Computer Software and Applications Conference, COMPSAC Workshops 2016, Atlanta, GA, USA, June 10-14, 2016* (2016) pp. 583–588.

[31] P. Samarati and L. Sweeney, *Generalizing data to provide anonymity when disclosing information (abstract),* in *Proceedings of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 1-3, 1998, Seattle, Washington, USA* (1998) p. 188.

[32] A. Inan, M. Kantarcioglu, and E. Bertino, *Using anonymized data for classification,* in *Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29 2009 - April 2 2009, Shanghai, China* (2009) pp. 429–440.

[33] C. Tai, P. S. Yu, and M. Chen, *k-support anonymity based on pseudo taxonomy for outsourcing of frequent itemset mining,* in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010* (2010) pp. 473–482.

[34] F. Giannotti, L. V. S. Lakshmanan, A. Monreale, D. Pedreschi, and W. H. Wang, *Privacy-preserving mining of association rules from outsourced transaction databases,* IEEE Systems Journal **7**, 385 (2013).

[35] C. C. Aggarwal and P. S. Yu, *A condensation approach to privacy preserving data mining,* in *Advances in Database Technology - EDBT 2004, 9th International Conference on Extending Database Technology, Heraklion, Crete, Greece, March 14-18, 2004, Proceedings* (2004) pp. 183–199.

[36] L. Qiu, Y. Li, and X. Wu, *Protecting business intelligence and customer privacy while outsourcing data mining tasks,* Knowl. Inf. Syst. **17**, 99 (2008).

[37] R. Chen, N. Mohammed, B. C. M. Fung, B. C. Desai, and L. Xiong, *Publishing set-valued data via differential privacy,* PVLDB **4**, 1087 (2011).

[38] R. Bhaskar, S. Laxman, A. D. Smith, and A. Thakurta, *Discovering frequent patterns in sensitive data,* in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010* (2010) pp. 503–512.

[39] L. Bonomi and L. Xiong, *A two-phase algorithm for mining sequential patterns with differential privacy,* in *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013* (2013) pp. 269–278.

[40] A. Friedman and A. Schuster, *Data mining with differential privacy,* in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010* (2010) pp. 493–502.

[41] N. Mohammed, R. Chen, B. C. M. Fung, and P. S. Yu, *Differentially private data release for data mining,* in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011* (2011) pp. 493–501.

[42] K. Lin, *Privacy-preserving kernel k-means outsourcing with randomized kernels,* in *13th IEEE International Conference on Data Mining Workshops, ICDM Workshops, TX, USA, December 7-10, 2013* (2013) pp. 860–866.

[43] K. Chen, G. Sun, and L. Liu, *Towards attack-resilient geometric data perturbation,* in *Proceedings of the Seventh SIAM International Conference on Data Mining, April 26-28, 2007, Minneapolis, Minnesota, USA* (2007) pp. 78–89.

[44] M. Kantarcioglu and C. Clifton, *Privacy-preserving distributed mining of association rules on horizontally partitioned data,* IEEE Trans. Knowl. Data Eng. **16**, 1026 (2004).

[45] T. Tassa, *Secure mining of association rules in horizontally distributed databases,* IEEE Transactions on Knowledge and Data Engineering **26**, 970 (2014).

[46] J. Vaidya and C. Clifton, *Secure set intersection cardinality with application to association rule mining,* Journal of Computer Security **13**, 593 (2005).

[47] B. Rozenberg and E. Gudes, *Association rules mining in vertically partitioned databases,* Data Knowl. Eng. **59**, 378 (2006).

[48] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, *Tools for privacy preserving data mining,* SIGKDD Explorations **4**, 28 (2002).

[49] Y. Lindell and B. Pinkas, *Secure multiparty computation for privacy-preserving data mining,* IACR Cryptology ePrint Archive **2008**, 197 (2008).

[50] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, *Secure knn computation on encrypted databases,* in *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2009, Providence, Rhode Island, USA, June 29 - July 2, 2009* (2009) pp. 139–152.

[51] X. Yi, F. Rao, E. Bertino, and A. Bouguettaya, *Privacy-preserving association rule mining in cloud computing,* in *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '15, Singapore, April 14-17, 2015* (2015) pp. 439–450.

[52] G. Tillem, Z. Erkin, and R. L. Lagendijk, *Mining encrypted software logs using alpha algorithm,* in *Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017) - Volume 4: SECRYPT, Madrid, Spain, July 24-26, 2017.* (2017) pp. 267–274.

[53] A. Burattin, M. Conti, and D. Turato, *Toward an anonymous process mining,* in *3rd International Conference on Future Internet of Things and Cloud, FiCloud 2015, Rome, Italy, August 24-26, 2015* (2015) pp. 58–63.

[54] T. E. Gamal, *A public key cryptosystem and a signature scheme based on discrete logarithms,* IEEE Trans. Information Theory **31**, 469 (1985).

[55] R. Cramer, R. Gennaro, and B. Schoenmakers, *A secure and optimally efficient multi-authority election scheme,* in *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding* (1997) pp. 103–118.

[56] Y. Lindell, *How to simulate it - A tutorial on the simulation proof technique,* in *Tutorials on the Foundations of Cryptography.* (Springer, 2017) pp. 277–346.

[57] C. Hazay and Y. Lindell, *Efficient Secure Two-Party Protocols - Techniques and Constructions*, Information Security and Cryptography (Springer, 2010).

[58] E. Barker and Q. Dang, *NIST Special Publication 800–57 Part 1, Revision 4*, Tech. Rep. (NIST, 2016).

[59] W. Steeman, *Bpi challenge 2013,* (2013).

4

PRIVACY-PRESERVING CONFORMANCE CHECKING FOR INTERNAL AUDITING

Organizations often use data analytic techniques to automate internal auditing tasks and monitor their processes. Conformance checking is an example technique, which checks whether the monitored behavior (recorded in an event log) complies with the normative behavior represented as a process model. Using such techniques, however, requires domain-specific knowledge and investment in resources and time, which cannot be afforded by every organization. Outsourcing audit tasks to specialized companies is thus becoming a common practice for many organizations to overcome the cost of in-house auditing. However, the disclosure of audit data to external parties raises serious privacy risks since this data often contains sensitive information. In this chapter, we propose two approaches for conformance checking under privacy preservation, which enables an organization to outsource detecting nonconformity in process executions while assuring the confidentiality of event logs. Our proposals rely on secure two-party computation to protect and process the event logs. We evaluated their performance using two real-life datasets, achieving promising results. To the best of our knowledge, our proposal is the first attempt that performs conformance checking under privacy preservation.

This chapter is based on the paper "*Privacy-Preserving Conformance Checking for Internal Auditing*" by G. Tillem, N. Zannone, and Z. Erkin (in preparation).

4.1. INTRODUCTION

The necessity of providing faster and localized services to customers requires technology companies to operate with multiple headquarters and offices around the globe. Although providing localized services simplifies the delivery of services to end-users, the maintenance of businesses becomes challenging with the increasing number of employees and the increasing complexity of processes. To verify the quality of operations in large scale companies, internal auditing is thus becoming an essential task. Internal audits aim to verify whether processes are executed within certain boundaries and legal requirements (such as Sarbanes-Oxley Act[1]), which is crucial for public accountability and defense against legal accusations [1]. A prominent example of using internal audits for public accountability is Google's investigation on employee salaries to defend the company against the accusations of wage discrimination based on gender [2].

Due to the obvious benefits of conducting systematic auditing, more and more companies have started employing data analytics tools to support their audit tasks [3]. Process mining provides a comprehensive solution to auditing by covering the whole process and data [1]. In particular, conformance checking has been proposed to evaluate the relation between an event log and a process model [4]. In this work, we focus on alignment-based conformance checking as alignments provide a robust approach to conformance checking by detecting the commonalities and deviations between the behavior recorded in the process model and the behavior recorded in the event log [5].

Although the use of data analytics such as alignment-based conformance checking allows automating the audit process, achieving high quality auditing is not trivial. One difficulty for companies is the lack of expertise that is required for special audit tasks [6]. Especially with the involvement of data analytics in audit processes, companies are in need of personnel specialized in the usage of analytics [7]. The availability of resources is also a main difficulty in auditing [6]. Shortage of employees, computational, or storage resources can hinder the audit process for companies. An approach to overcome these problems is to outsource audit tasks to external specialized parties [8]. Outsourcing audit tasks reduces the need for resources while assuring the necessary expertise.

As pointed out in [6], although outsourcing relieves companies from the burden of in-house audit, it introduces privacy concerns associated with the disclosure of sensitive information (e.g., customer and employee data). Especially with the introduction of the GDPR in the EU, organizations have become even more cautious in data sharing due to the legal implications of data misuse. Similarly, the leakage of the corporate information, which might be inferred from logs, can threaten the intellectual property of organizations. To mitigate these risks, there is a need of solutions that allow organizations to outsource their audit tasks while protecting the confidentiality of log data.

Privacy considerations in outsourced data is a popular research area, where many solutions based on statistical and cryptographic mechanisms are proposed to perform well-known data analytics techniques. A group of solutions use differential privacy [9] to anonymize data in mining frequent patterns [10, 11], learning decision trees [12], or training neural networks [13]. However, these approaches are not suitable for the analysis of process executions as they are not able to account for the control-flow perspective,

[1] http://www.soxlaw.com

which is essential for the analysis of processes. Moreover, the accuracy of the analysis might be degraded due to the noise added to ensure anonymity. Another approach to protect the confidentiality of sensitive data is the use of homomorphic encryption (HE), which enables computation of arithmetic operations on encrypted data without intermediate decryption [14]. Several solutions based on HE have been proposed for data mining and machine learning [15–18]. However, these solutions suffer from high computation costs due to the expansion in data size. An efficient alternative to HE is secure multiparty computation (MPC), where multiple parties compute a function on their private inputs without revealing any information about their inputs [19]. MPC enables to design interactive solutions that are not restricted to arithmetic operations, yet, significantly more efficient in computation cost compared to HE-based alternatives [20–22].

In this paper, we propose a solution for privacy-preserving alignment-based conformance checking that enables an organization to outsource the analysis of its log for detecting nonconformity in process executions while assuring the confidentiality of log data. Our solution operates in a server-aided setting, in which an auditee outsources its computations to two non-colluding auditors who are responsible to perform operations and return the result to the auditee. This setting is well-known for privacy-preserving data analytics since it significantly reduces the computation cost on the auditee side [22–25]. We choose secure multiparty computation as the underlying privacy-preserving technology since it achieves an efficient computation performance compared to HE-based alternatives, and maintains the accuracy of computations in contrast to differential privacy. Under the semi-honest security assumption, our solution guarantees the confidentiality of log data while process diagnostics is only revealed to the data owner. To the best of our knowledge, our solution is the first proposal that performs conformance checking under privacy preservation.

Our contribution can be summarized as follows:

- We propose two protocols for secure alignment-based conformance checking.
  - Our first protocol – **SCORCH$_{\mathbf{EXH}}$** – guarantees privacy using an exhaustive approach. It computes all possible alignments between the event log and the process model under privacy preservation.
  - Our second protocol – **SCORCH$_{\mathbf{PQ}}$** – provides an efficient alternative to the first protocol by reducing computation and communication complexity. It uses a priority queue, which eliminates the need of computing all possible alignments. However, reducing the number of computed alignments might weaken the security of the protocol. To minimize the information that can be inferred from the computation, we set a threshold for the number of alignments to be computed.
- We show that our protocols are secure with respect to the information theoretical security of secure multiparty computation.
- We evaluate the feasibility of our protocols on two real-life event logs [26, 27]. The results show that our protocols can perform conformance checking with reasonable computation and communication costs while assuring the confidentiality of log data.

The remainder of the paper is organized as follows. The next section provides preliminary knowledge on conformance checking and on the cryptographic primitives. Section 4.3 presents our protocols and Section 4.4 analyzes their security. Section 4.5 presents an experimental evaluation of our protocols. Section 4.6 discusses related work and Section 4.7 concludes the paper.

4.2. PRELIMINARIES

This section provides preliminary knowledge on conformance checking and the cryptographic techniques used for the design of our protocols.

4.2.1. CONFORMANCE CHECKING

Conformance checking aims to verify whether the observed behavior recorded in an event log matches the intended behavior represented as a process model. In this work, we consider process models in the form of Labeled Petri nets.

Definition 4.2.1 (Labeled Petri Net)**.** A Labeled Petri net is a tuple $(P, T, F, A, \ell, m_i, m_f)$ where P is a set of places; T is a set of transitions; $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation between places and transitions (and between transitions and places); A is the set of labels for transitions; $\ell : T \to A$ is a function that associates a label with every transition in T; m_i is the initial marking; m_f is the final marking.

The label of a transition identifies the *activity* represented by such a transition. Multiple transitions can be associated with the same activity label. Some transitions can be invisible. Invisible transitions do not correspond to actual activities but are used for routing purposes and, thus, their execution is never recorded in event logs. Given a Petri net N, $Inv_N \subseteq A$ indicates the set of labels associated to invisible transitions. Following convention, given a node $x \in P \cup T$, we denote its *preset* (i.e., the set of its input nodes) as ${}^\bullet x = \{x' \in P \cup T \mid (x', x) \in F\}$ and its postset (i.e., the set of its output nodes) as $x^\bullet = \{x' \in P \cup T \mid (x, x') \in F\}$.

The state of a Petri net is represented by a multi-set of tokens on the places of the net, i.e. the *marking*. A transition is enabled if all its input places contain at least one token. When an enabled transition is fired, a token is taken from each of its input places and a token is added to all its output places. A *process run* is a sequence of (instances of) transitions such that firing the transitions in the sequence from the initial marking m_i leads to a proper termination state, i.e. to the final marking m_f. Ψ_N denotes the set of all process runs allowed by N. Figure 4.1 shows an example Petri net that visualizes the Italian road fine management process [27].

The execution of activities is often logged by information systems. A logged instance of an activity is called *event*.

Definition 4.2.2 (Event, Attribute)**.** Let $\mathscr{E}$ be the set of all possible events. An event can have attributes. Let $\mathscr{U}$ be the attribute universe. For all events $e \in \mathscr{E}$ and attribute $att \in \mathscr{U}$, $\pi_{att}(e)$ is the value of *att* for e.

In this work, we assume that (instances of) transitions and events have at least the following attributes: *case*, for the process instance to which the event belongs; *act*, for the activity associated to the event; *time*, for the timestamp of the event.

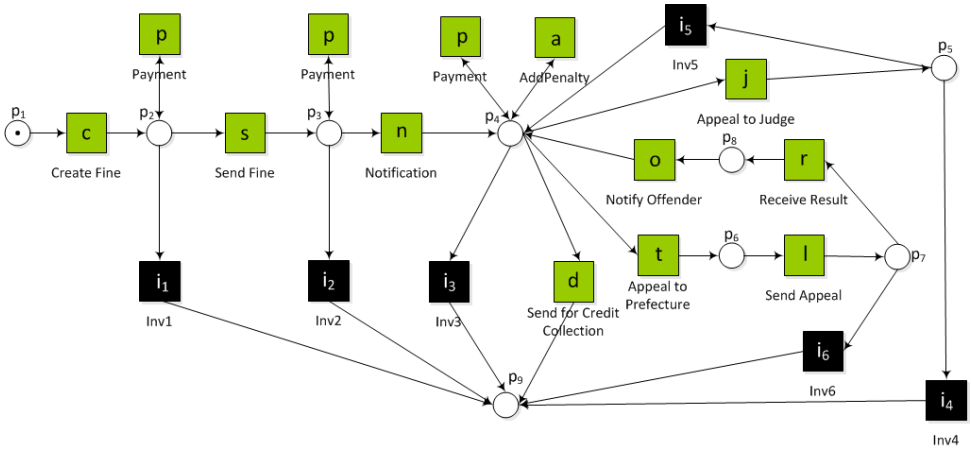


Figure 4.1: A sample process model representing the management of road traffic fines. Green boxes represent transitions associated to activities and black boxes invisible transitions. The text below the transitions represent the label, which is shortened with a single letter inside the transitions.

Definition 4.2.3 (Event Trace, Event Log)**.** An event trace $\sigma = \langle e_1, \ldots, e_n \rangle \in \mathscr{E}^*$ is a finite sequence of events $e_1, \ldots, e_n \in \mathscr{E}$ recorded for a process instance (case) such that, for every $e_i, e_j \in \sigma$ with $1 \leq i < j \leq n$, (1) $e_i \neq e_j$ and (2) $\pi_{time}(e_i) \leq \pi_{time}(e_j)$. An event log L is a finite set of event traces.

Table 4.1 shows some sample traces from the road traffic fine management log. Each trace is associated with a case id. Each event within a trace has several attributes that include activity name, timestamp, the officer id associated with the event and the payment amount for the corresponding event.

Most of the state-of-the-art techniques for computing the conformance of event traces with a process model are based on the notion of alignment [5]. Alignments provide a robust approach to conformance checking, which is able to pinpoint deviations causing nonconformity. If an event trace perfectly fits the net, each "move" in the event trace can be mimicked by a "move" in the model, i.e. a transition fired in the net. After all events in the event trace are mimicked, the net reaches its final marking. In cases where deviations occur, some moves in the event trace cannot be mimicked by the process model or vice versa. We explicitly denote "no move" by $\gg$.

Given a Petri net $N = (P, T, F, A, \ell, m_i, m_f)$ and an event trace σ, let $S_M = A \cup \{\gg\}$ and $S_L = \mathscr{E}_\sigma \cup \{\gg\}$ with $\mathscr{E}_\sigma$ the set of events occurring in σ. A legal move for σ and N is a pair $(m_L, m_M) \in (S_L \times S_M) \setminus \{(\gg, \gg)\}$ such that

- (m_L, m_M) is a *synchronous move* if $m_L \in \mathscr{E}_\sigma$, $m_M \in A$ and $\pi_{act}(m_L) = m_M$,
- (m_L, m_M) is a *move on log* if $m_L \in \mathscr{E}_\sigma$ and $m_M = \gg$,
- (m_L, m_M) is a *move on model* if $m_L = \gg$ and $m_M \in A$.

$\Sigma_{(\sigma,N)}$ denotes the set of legal moves for an event trace σ and a Petri net N.

Table 4.1: Sample traces from the road traffic fine management event log.

Case ID	Event Attributes					
	ID	Activity	Timestamp	Officer ID	Amount	⋯
σ_1	e_1	Create fine (c)	02.08.2006	541	35.0	⋯
	e_2	Send fine (s)	12.12.2006	561		⋯
	e_3	Notification (n)	15.01.2007	557		⋯
	e_4	Add Penalty (a)	16.03.2007	550	71.5	⋯
	e_5	Send for credit coll. (d)	30.03.2009	537		⋯
σ_2	e_1	Create fine (c)	19.03.2007	541	36.0	⋯
	e_2	Send fine (s)	17.07.2007	561		⋯
	e_3	Notification (n)	25.07.2007	557		⋯
	e_4	Appeal to prefecture (t)	02.08.2007	558	74.0	⋯
	e_5	Add penalty (a)	23.09.2007	550		⋯
	e_6	Send appeal (l)	24.09.2007	559		⋯
σ_3	e_1	Create fine (c)	20.03.2007	561	22.0	⋯
	e_2	Send fine (s)	17.07.2007	541		⋯
	e_3	Notification (n)	23.07.2007	557		⋯
	e_4	Add Penalty (a)	21.09.2007	550	44.0	⋯
	e_5	Payment (p)	01.10.2007	550		⋯
	e_6	Payment (p)	31.10.2007	550		⋯
σ_4	e_1	Create fine (c)	24.07.2006	541	35.0	⋯
	e_2	Send fine (s)	05.12.2006	561		⋯
σ_5	e_1	Create fine (c)	11.09.2006	561	21.0	⋯
	e_2	Send fine (s)	29.12.2006	561		⋯
	e_3	Notification (n)	09.01.2007	557		⋯
	e_4	Add Penalty (a)	10.03.2007	550	42.5	⋯
	e_5	Send appeal (l)	05.04.2007	559		⋯

Definition 4.2.4 (Alignment)**.** Let $\Sigma_{(\sigma,N)}$ be the set of legal moves for a Petri net $N = (P, T, F, A, \ell, m_i, m_f)$ and event trace σ. An alignment of σ and N is a sequence $\gamma \in \Sigma^*_{(\sigma,N)}$ such that, ignoring all occurrences of $\gg$, the projection on the first element yields σ and the projection on the second element yields a sequence $\langle a_1, \ldots, a_n \rangle$ such that there exists a firing sequence $\psi' = \langle t_1, \ldots, t_n \rangle \in \text{prefix}(\psi)$ for some $\psi \in \Psi_N$ where, for every $1 \leq i \leq n$, $\pi_{act}(t_i) = a_i$. If $\psi' \in \Psi_N$, γ is called a complete alignment of σ and N.

There can exist multiple (possibly infinite) alignments for a given event trace and process model. Figure 4.2 illustrates different possible alignments between the event trace $\sigma_5 = \langle c, s, n, a, l \rangle$ in Table 4.1 and the model in Figure 4.1. The top row of an alignment shows the sequence of activities in the event trace, and the bottom row shows the sequence of activities in the process model (both ignoring $\gg$).

To determine the quality of alignments, a cost is assigned to each move in the alignment. The cost is defined as the sum of the costs of the individual moves in the alignment. An *optimal alignment* of an event trace and a process model is one of the align-

$$\gamma_1 = \begin{array}{|c|c|c|c|c|c|c|} \hline c & s & n & a & \gg & l & \gg \\ \hline c & s & n & a & t & l & i_6 \\ \hline \end{array} \qquad \gamma_2 = \begin{array}{|c|c|c|c|c|c|} \hline c & s & n & a & l & \gg \\ \hline c & s & n & a & \gg & d \\ \hline \end{array}$$

$$\gamma_3 = \begin{array}{|c|c|c|c|c|c|c|c|c|} \hline c & s & n & a & \gg & l & \gg & \gg & \gg \\ \hline c & s & n & a & t & l & r & o & i_3 \\ \hline \end{array}$$

Figure 4.2: Sample alignments between σ_5 and the model in Figure 4.1

ments with the lowest cost for the given cost function. As an example, consider a cost function that penalizes all moves on model for visible transitions and moves on log (i.e., it assigns cost 1 to those moves and 0 otherwise). If moves on model for invisible transitions i_k are ignored, γ_1 has one move on model, γ_2 has one move on model and one move on log, and γ_3 has three moves on model. Since there does not exist an alignment with lower cost, γ_1 is an optimal alignment for σ_5 and the process model.

4.2.2. SECURE TWO-PARTY COMPUTATION

Secure two-party computation enables two parties to evaluate a function f jointly on their private inputs x, y without revealing any information about their inputs apart from the result of the function $f(x, y)$ [28]. Since its introduction [29], secure two-party computation has been largely investigated and practical frameworks implementing it have been developed [30–32].

We implement secure two-party computation protocols using the ABY framework in [30]. ABY contains three types of secure two-party circuits: arithmetic circuits [33], Boolean circuits [19], and Yao's garbled circuits [29]. In our work, we use only arithmetic circuits and Boolean circuits. Given parties P_1 and P_2, ABY creates secret shares for each party, denoted $[\cdot]_i^A$ for arithmetic circuits and $[\cdot]_i^B$ for Boolean circuits (with $i \in \{1,2\}$), and evaluates f on these shares. The shares are created as follows:

- **Arithmetic circuit:** Let x be an ℓ-bit value, the arithmetic shares of parties P_1, P_2 are $[x]_0^A, [x]_1^A$ such that $[x]_0^A + [x]_1^A = x \mod 2^\ell$. The parties can compute addition and multiplication on their private ℓ-bit inputs x and y as follows:
  - addition of $[x]^A$ and $[y]^A$ is $[z]^A \leftarrow [x]^A + [y]^A \mod 2^\ell$,
  - multiplication of $[x]^A$ and $[y]^A$ is $[z]^A \leftarrow [x]^A \times [y]^A \mod 2^\ell$.
- **Boolean circuit:** Let x be a single bit value, the Boolean shares of parties P_1, P_2 are $[x]_0^B, [x]_1^B$ such that $[x]_0^B \wedge [x]_1^B = x \mod 2$, where $\wedge$ represents the bitwise XOR operation. ABY implements XOR and AND gates for Boolean circuits, which is sufficient to design every efficiently computable function [30]. We use Boolean circuits to compute the following operations on the private ℓ-bit inputs x and y:
  - addition of $[x]^B$ and $[y]^B$ such that $[z]^B \leftarrow [x]^B + [y]^B \mod 2^\ell$,
  - multiplication of $[x]^B$ and $[y]^B$ such that $[z]^B \leftarrow [x]^B \times [y]^B \mod 2^\ell$,

- comparison of $[x]^B$ and $[y]^B$ such that $[z]^B \leftarrow [x]^B \overset{?}{>} [y]^B$, where $z = 1$ if $x > y$ and 0 otherwise,
- equality of $[x]^B$ and $[y]^B$ such that $[z]^B \leftarrow [x]^B \overset{?}{=} [y]^B$, where $z = 1$ if $x = y$ and 0 otherwise.

In the ABY framework, performing additions and multiplications with arithmetic circuits are much cheaper than the ones with Boolean circuits. Thus, in our implementation we use arithmetic circuits to perform additions and multiplications, and Boolean circuits for the other operations. ABY provides conversion functions between Boolean sharing and arithmetic sharing such that $B2A$ is a conversion function from Boolean to arithmetic sharing, and $A2B$ is a conversion function from arithmetic to Boolean sharing.

4.3. SCORCH: SECURE CONFORMANCE CHECKING

In this section, we present two protocols to securely compute optimal alignments. Our first protocol, $\text{SCORCH}_{\text{EXH}}$, computes alignments exhaustively while our second protocol, $\text{SCORCH}_{\text{PQ}}$ computes the alignments using a private priority queue.

4.3.1. SETTING AND THREAT MODEL

We consider a server-aided setting such that a client outsources its operations to two non-colluding semi-honest servers who perform computations together. Our scenario contains three entities, which are an auditee and two non-colluding semi-honest audit companies. The auditee is a company who lacks expertise or resources for auditing, thus, outsources the audit task to external parties. To compensate its limited computational power, auditee delegates the computations to two non-colluding auditors. More specifically, the auditee shares a protected event log with the auditors that contains the workflow information of the company. The goal of auditors is to securely check if the employees of auditee conform with the prescribed behavior (i.e. the process model) by computing the alignments between the process model and the event log.

The inputs of our protocols are a protected query $[\sigma]^A$ and a protected Petri net $[N]^A$, which are created by the auditee using arithmetic sharing and sent to the auditors before the start of execution. The protection assures the activity label of places in $[N]^A$ and activity label of events in $[\sigma]^A$ are confidential. However, the number of events in $[\sigma]^A$, the number of places in $[N]^A$ are not hidden from the auditors. The output is revealed to an authorized entity only if there is a nonconformity between $[\sigma]^A$ and $[N]^A$. The nonconformity is detected by checking if the alignment cost is greater than zero.

In the design of a privacy-preserving conformance checking protocol, we aim at several goals related to correctness, privacy, and efficiency. Our protocols should:

- correctly compute optimal alignments for each protected trace,
- protect the content of the trace from the auditors,
- protect the output and its length from the auditors,
- hide the execution order of the places to prevent tracking on Petri net,

- be feasible in computation and communication cost,
- minimize the computations on the auditee's side.

Existing techniques for privacy preservation enable us to protect the content of the trace and the output from auditors while achieving a realistic computation and communication cost. We can also minimize the auditee's computation cost by using a server-aided scenario. However, hiding the execution order of places is not a trivial task. There exists several techniques to prevent tracking, such as private information retrieval (PIR) [34] or oblivious RAM (ORAM) [35]. Considering their expensive nature, using PIR/ORAM repetitively might increase the computation and communication cost drastically. Therefore, to hide the execution order of the places in an efficient way, we design two protocols such that the first protocol computes the solution exhaustively to cover all possible access patterns; the second protocol computes the solution using a private priority queue, which eliminates the need for PIR/ORAM operations.

4.3.2. ALIGNMENT NODE

Computing optimal alignments requires maintaining information about alignment sequences. To handle this information, we introduce a node structure, *alignment node*, which stores the information in every step of the protocol. These nodes are organized in a tree structure, where every node has at most three child nodes. An alignment node n has:

- an index (i, j) that is assigned to each node such that i represents the depth of the node in the tree and j represents the order of the node in depth i,
- a cost c, which is the cost of the alignment in the current node n, and
- an alignment sequence γ, which stores the alignment between the Petri net N and (a prefix of) the event trace σ.

A node has at most three children nodes which are n^L for a move on log, n^M for a move on model, and n^S for either a synchronous move ($\pi_{act}(m_L) = m_M$) or an illegal move ($\pi_{act}(m_L) \neq m_M$). In Figure 4.3 we represent an alignment node n with its children. The indices of child nodes are set w.r.t. the index of the parent node. The alignment sequence and its cost are determined w.r.t. the relation between activity labels of events $a_\sigma = \pi_{act}(e_a)$, s.t. $e_a \in \sigma$, and activity labels of transitions $a_N = \ell(t_b)$, s.t. $t_b \in T$.

The cost of n^M and n^L are always greater than the cost of parent node n by 1 when the transition is not invisible. In case of an invisible transition, the cost of n^M remains the same. The cost of n^S depends on the relation between a_σ and a_N. If the two activities are equal then the cost is equal to the cost of $n_{i,j}$ since it is a synchronous move. Otherwise, the cost is set to $n_{i,j}.c + \text{INT\_MAX}$. The alignment sequence γ is updated for each child with an append operation ($\oplus$). For n^M, γ becomes $n_{i,j}.\gamma \oplus (a_\sigma, \gg)$ since only the activity label of the trace is executed. For n^L, γ is $n_{i,j}.\gamma \oplus (\gg, a_N)$ since only the activity label of the Petri net is executed. For n^S, γ becomes $n_{i,j}.\gamma \oplus (a_\sigma, a_N)$ since both the corresponding activities were executed. Additionally, a node can have several flags such as n.endN and n.endT, which are used to signal the end of activities from N or σ, respectively. When an alignment node is created these flags are set to *false*. If N reaches to its final state, then n.endN is set to *true*. If all events in σ are executed, then n.endT is set to $true$.

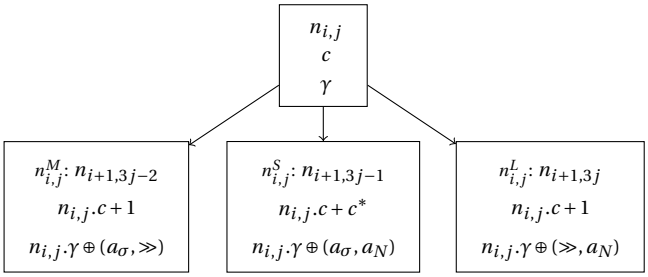


Figure 4.3: Alignment node. For n^S if $a_\sigma = a_N$, then $c^* = 0$, otherwise $c^* =$ INT_MAX.

4

4.3.3. SCORCH$_{\text{EXH}}$: SECURE CONFORMANCE CHECKING VIA EXHAUTIVE SEARCH

Our first protocol computes an optimal alignment n_{OPT} exhaustively by generating all possible alignments. The input of the protocol is a trace $[\sigma]^A$ and a Petri Net $[N]^A$. We summarize the flow of the protocol in Algorithm 8. The iteration of the protocol is recursive. It starts with a null node n, recursively calls functions CALCULATE_NEXT, MOVE_ON_MODEL, MOVE_SYNCHRONOUS, and MOVE_ON_LOG until all possible alignments are computed. In each iteration, CALCULATE_NEXT function checks if N and σ are completed. If they are both completed, then it performs a secure comparison between n_{OPT} and the new alignment. Since the secure comparison is performed under Boolean sharing, the sharing type should be switched from arithmetic to Boolean beforehand. If either σ or N (or none of them) have not been completed, then the move functions, i.e. MOVE_ON_MODEL, MOVE_SYNCHRONOUS, and MOVE_ON_LOG, are executed (lines 7-14 in Algorithm 8). It is worth noting that, in the move functions, the cost and the alignment of the child nodes are computed under privacy preservation using secure addition and multiplication. The indices and the flags n.endT and n.endN are kept as plaintext since the algorithm generates all possibilities with a deterministic order.

In SCORCH$_{\text{EXH}}$, we assume no loops in N. Since the protocol considers all possible process runs allowed by the process model, the presence of loops in the model would lead to the generation of an infinite number of process runs, and the protocol would not terminate. Furthermore, during the protocol execution, if there is more than one element in the postset or preset of a place or a transition, the protocol has to choose one of the elements to continue the next move. We handle this issue by computing MOVE_ON_MODEL and MOVE_SYNCHRONOUS functions for each possible elements in the postset or preset. This operation is not needed in MOVE_ON_LOG since the move is performed only on the trace, not on the process model.

Figure 4.4 illustrates the nodes generated by SCORCH$_{\text{EXH}}$ for trace σ_5 in Table 4.1. Starting from the root node, the protocol generates a tree using the recursive functions. The sub-trees that are connected by red dashed lines show the clones created during the execution when there are more than one elements in the preset or postset of a transition. Once all possible alignments have been computed, the protocol returns the optimal alignment, which is highlighted in the figure with a green border.

Algorithm 8 $\text{SCORCH}_{\text{EXH}}([\sigma]^A, [N]^A)$

1: $n \leftarrow null$, $n_{OPT} \leftarrow null$, $n_{OPT}.c \leftarrow$ INT_MAX , n.endN = *false* , n.endT = *false*
2: $n_{OPT} \leftarrow$ CALCULATE_NEXT(n)
3: **return** n_{OPT}
4: **procedure** CALCULATE_NEXT(n)
5: **if** n.endN = *true* **and** n.endT = *true* **then**
6: $n_{OPT} \leftarrow (n_{OPT}.[c]^B > n.[c]^B)\ ?\ n : n_{OPT}$
7: **else if** n.endN = *true* **and** n.endT = *false* **then**
8: MOVE_ON_LOG(n)
9: **else if** n.endN = *false* **and** n.endT = *true* **then**
10: MOVE_ON_MODEL(n)
11: **else**
12: MOVE_ON_MODEL(n)
13: MOVE_SYNCHRONOUS(n)
14: MOVE_ON_LOG(n)
15: **return** n_{OPT}
16: **procedure** MOVE_ON_MODEL(n)
17: **for all** $|N.^\bullet x|$ **or** $|N.x^\bullet|$ **do**
18: $n^M_{i,j} \leftarrow n_{i+1,3j-2}$, n^M.endT $\leftarrow n$.endT
19: $n^M.[c]^A \leftarrow n.[c]^A + [1]^A$
20: $n^M.[\gamma]^A \leftarrow n.[\gamma]^A \oplus ([\gg]^A, [a_N]^A)$
21: **if** N completed **then** n^M.endN $\leftarrow true$
22: CALCULATE_NEXT(n^M)
23: **procedure** MOVE_SYNCHRONOUS(n)
24: **for all** $|N.^\bullet x|$ **or** $|N.x^\bullet|$ **do**
25: $n^S_{i,j} \leftarrow n_{i+1,3j-1}$
26: $n^S.[c]^A \leftarrow n.[c]^A + ([a_N]^A - [a_\sigma]^A) \cdot [\text{INT\_MAX}]^A$
27: $n^S.[\gamma]^A \leftarrow n.[\gamma]^A \oplus ([a_\sigma]^A, [a_N]^A)$
28: **if** N completed **then** n^S.endN $\leftarrow true$
29: **if** $[\sigma]^A$ completed **then** n^S.endT $\leftarrow true$
30: CALCULATE_NEXT(n^S)
31: **procedure** MOVE_ON_LOG(n)
32: $n^L_{i,j} \leftarrow n_{i+1,3j}$, n^L.endN $\leftarrow n$.endN
33: $n^L.[c]^A \leftarrow n.[c]^A + [1]^A$
34: $n^L.[\gamma]^A \leftarrow n.[\gamma]^A \oplus ([a_\sigma]^A, [\gg]^A)$
35: **if** $[\sigma]^A$ completed **then** n^L.endT $\leftarrow true$
36: CALCULATE_NEXT(n^L)

COMPLEXITY ANALYSIS

Given the length k of the trace and the length ℓ of longest path in the Petri net, the complexity of $\text{SCORCH}_{\text{EXH}}$ is $\mathcal{O}(3^{k+\ell})$. It is important to remind that for our first protocol, we consider loop-free processes. In the existence of loops, the length of the longest process run becomes infinite in theoretical bounds. The overhead of the privacy preservation in

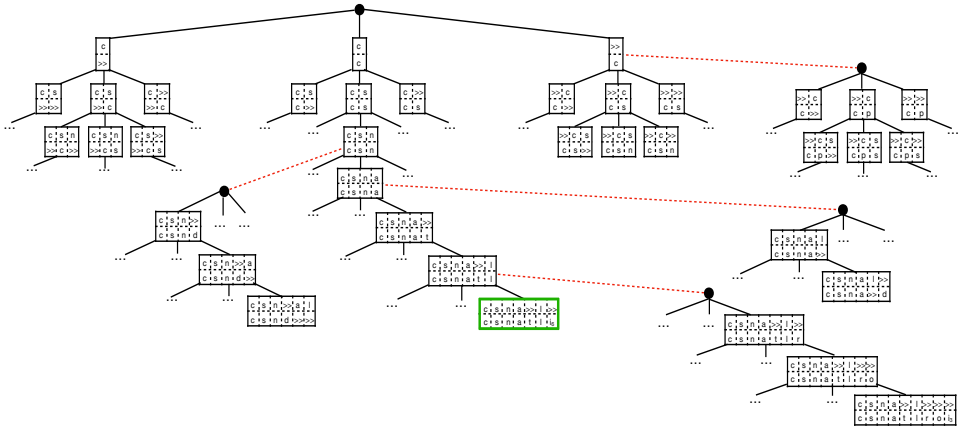


Figure 4.4: Execution of Protocol 8 on $\sigma_5 = \langle c, s, n, a, l \rangle$. The optimal alignment is outlined in green. Red dashed lines show the cloning points for multiple elements in the postset or preset.

computation is caused by secure addition, multiplication and comparison. Secure addition can be performed locally by each auditor; thus, it does not have any additional computation cost. Performing multiplication and comparison is not straightforward under privacy preservation since it adds several rounds of computation and communication on the original operation. However, it is important to note that the number of secure multiplication and secure comparison operations are much less than secure additions. Secure multiplication is only performed in synchronous move operation, and secure comparison is only performed on the leaf nodes.

4.3.4. SCORCH$_{PQ}$: SECURE CONFORMANCE CHECKING VIA PRIORITY QUEUES

Exhaustively computing all possible alignments is successful in hiding the length of the output and the execution order of places since all possible options are covered. However, using an exhaustive approach is not desired because of the high computation cost and memory usage. As an alternative, we design a second protocol, SCORCH$_{PQ}$, that uses a priority queue to keep track of the alignment with minimum cost. The use of a priority queue enables us to compute the optimal alignment with less number of iterations. In SCORCH$_{PQ}$, the auditee should interrupt the protocol to finish the execution. The reason is that all values are computed privately and, thus, the auditors cannot observe when an optimal alignment is found. On the other hand, terminating the protocol in less number of iterations might leak information about the length of the output and the alignment cost. Therefore, we limit the auditee's interruption to a certain interval, T, to check if an optimal alignment has been found. In the worst case, the protocol has the same complexity as the first protocol.

Algorithm 9 provides an overview of SCORCH$_{PQ}$. In addition to $[\sigma]^A$ and $[N]^A$, it requires a threshold T as input, which can be determined by the auditee with respect to the number of events in σ and the number of places in N. The protocol uses two private

priority queues Q and Q^R to keep track of the optimal alignment. Each element of the priority queues is an alignment node. Q is the main priority queue in which newly created alignment nodes are inserted. Q^R stores the nodes that are removed from Q. Before presenting the protocol, we briefly explain the functions of private priority queues.

Private Priority Queue: In a private priority queue, we denote the front node of the queue as Q_{front} and the rear node of the queue as Q_{rear}. The nodes are connected via a pointer called $next$. A private priority queue implements the following functions:

- ENQUEUE(n) adds a new node n to the rear such that $Q_{rear} = n$.
- DEQUEUE(n) removes the node n at the front such that $Q_{front} = n.next$.
- IS_EMPTY checks if Q_{front} and Q_{rear} are null.
- PRIVATE_REORDER reorders the nodes in Q such that the node with the least alignment cost is moved to the front.
- PRIVATE_REORDER_R reorders the nodes in Q^R such that the node with a completed alignment sequence (i.e. n.endT = *true* and n.endN = *true*) and with the least alignment cost is moved to the front.
- SWAP(n_1, n_2) takes two alignment nodes n_1 and n_2 as input and swaps their location in the queue privately.

The protocol starts by inserting an empty node n into Q and it iterates until Q is empty (lines 4-19 in Algorithm 9). An important remark is that in practice the queue never becomes empty since the protocol keeps generating new alignments although both σ and N are completely executed. However, as explained previously, with the interruption of the auditee, the protocol can terminate in a realistic number of iterations.

At every iteration, $n = Q_{front}$ is dequeued and it is enqueued into Q^R. If the number of iterations reach T or a multiple of T (line 8), then the protocol dequeues the front node of Q^R, i.e. n^R, and checks whether n^R corresponds to a completed alignment sequence, i.e. both σ and N are completely executed (line 10). If so, the protocol returns n^R as the optimal alignment and terminates. Otherwise, n^R is enqueued into Q^R and the execution continues with the move functions MOVE_ON_MODEL(n), MOVE_SYNCHRONOUS(n), MOVE_ON_LOG(n), and the private reordering of Q.

The move functions work slightly different than the ones described in Algorithm 8. At the end of each function, the newly created node is inserted into Q, instead of calling CALCULATE_NEXT(n) function. Another difference is deciding the next place in the execution. The information about the next place should be kept hidden since it can reveal the execution order of places. To hide the next place and the activity executed, we use private lookup tables. The tables are secret shared between the two auditors. By receiving the index value of the current place, the auditors can compute the next place and the activity associated with it using secure equality checks.

4.3.5. EXPLANATION OF LOOKUP TABLES

In this section, we provide more details about the move functions used in SCORCH$_{PQ}$. Different from the ones in SCORCH$_{EXH}$, the move functions in SCORCH$_{PQ}$ do not make

Algorithm 9 $\text{SCORCH}_{\text{PQ}}([\sigma]^A,\ [N]^A,\ T)$

1: $n \leftarrow null$, $n^R \leftarrow null$, $Q \leftarrow null$, $Q^R \leftarrow null$, $itr \leftarrow 0$
2: $n_{OPT} \leftarrow null$, $n_{OPT}.[c]^A \leftarrow [\text{INT\_MAX}]^A$
3: Q.ENQUEUE(n)
4: **while** Q.IS_EMPTY() = *false* **do**
5: Q.DEQUEUE(n)
6: Q^R.ENQUEUE(n)
7: Q^R.PRIVATE_REORDER_R()
8: **if** $itr \mod T = 0$ **then**
9: Q^R.DEQUEUE(n^R)
10: **if** *true* $\leftarrow$ reveal($n^R.[\text{endN}]^B \stackrel{?}{=} [true]^B$ **and** $n^R.[\text{endT}]^B \stackrel{?}{=} [true]^B$) **then**
11: $n_{OPT} \leftarrow n^R$
12: **return** n_{OPT}
13: **else**
14: Q^R.ENQUEUE(n^R)
15: MOVE_ON_MODEL(n)
16: MOVE_SYNCHRONOUS(n)
17: MOVE_ON_LOG(n)
18: Q.PRIVATE_REORDER()
19: $itr \leftarrow itr + 1$
20: **return** n_{OPT}
21: **procedure** PRIVATE_REORDER()
22: $n \leftarrow Q_{front}$, $n_{MIN} \leftarrow null$, $n_{MIN}.[c]^B \leftarrow [\text{INT\_MAX}]^B$
23: **while** $n \neq null$ **do**
24: $n_{MIN} \leftarrow \left(n_{MIN}.[c]^B > n.[c]^B\right)\ ?\ n : n_{MIN}$
25: $n \leftarrow n.next$
26: Q.SWAP(Q_{front}, n_{MIN})
27: **procedure** PRIVATE_REORDER_R()
28: $n \leftarrow Q^R_{front}$, $n_{MIN} \leftarrow null$, $n_{MIN}.[c]^B \leftarrow [\text{INT\_MAX}]^B$
29: **while** $n \neq null$ **do**
30: $n_{MIN} \leftarrow n.[\text{endN}]^B \stackrel{?}{=} [true]^B$ **and** $n.[\text{endT}]^B \stackrel{?}{=} [true]^B$ **and** $\left(n_{MIN}.[c]^B > n.[c]^B\right)\ ?\ n : n_{MIN}$
31: $n \leftarrow n.next$
32: Q^R.SWAP(Q^R_{front}, n_{MIN})
33: **procedure** SWAP(n_1, n_2)
34: $n_{tmp} \leftarrow n_1$
35: $n_1 \leftarrow n_2$
36: $n_2 \leftarrow n_1$

recursive calls to CALCULATE_NEXT function with the newly created alignment node, rather they insert the new node into Q. Furthermore, to prevent leakage of execution order of the nodes $x \in P \cup T$ on the Petri net, the functions use private look up tables

which securely computes:

- the next node to be executed x_{next},
- the activity label associated with the transition between the current node and the next node a_N,
- the activity label of the executed event in the trace a_σ,
- if the transition is invisible isInv,
- if the Petri net has reached its final state endN, and
- if the trace has completed endT.

To implement the secure look up operations, we introduce two more attributes to the alignment node. The first attribute x_{curr} is an index value that maintains information about the current transition to be executed. The second attribute is e_{curr}, which is the index of the current event in σ. We perform look up operations on the process model and on the trace in two functions separately. The first function LOOKUP_N receives $n.[x_{curr}]$ as input and computes $[x_{next}]$, $[a_N]$, [isInv], and [endN]. The second function LOOKUP_T takes $[e_{curr}] \in \sigma$ as input and computes $[a_\sigma]$ and [endT] privately.

The secure look up operation on N is performed using four look up tables which are IND, P^{LUT}, A^{LUT}, and I^{LUT}. IND is an index table that keeps track of the current node x_{curr}. If a node has multiple elements in its postset ($|x^\bullet| > 1$), then its index is repeated as the number of elements in the postset. P^{LUT} maintains information about transitions between the places of N. A^{LUT} stores the activity labels executed along with the transitions. Finally, I^{LUT} is a Boolean table that checks whether the executed transition is invisible.

In Table 4.2, we present the sample tables IND, P^{LUT}, A^{LUT}, and I^{LUT} for the process model in Figure 4.1. The process model has nine places which are represented as $p_1, p_2, \ldots, p_9$. In IND and P^{LUT} tables, we encode this places to indices $0, 1, \ldots, 8$. For instance, the first index in IND corresponds to p_1, for which the index value is 0, and the first index of P^{LUT} is 1, which corresponds to p_2 as the successor of p_1. Indices 1, 2, and 3 in IND represents p_2 with all elements in its postset, which are represented in the corresponding indices of P^{LUT} as $1, 2, 8$, i.e. $p_2,\ p_3,\ p_9$. A^{LUT} lists the activity labels that are associated with the current transition. In the implementation, we encode these string values as integers. Finally, in I^{LUT}, an index value is set to 1 if the transition associated with that index is invisible.

In Algorithm 10 we present the pseudocode for the private look up functions which are LOOKUP_N and LOOKUP_T. In function LOOKUP_N, apart from $[x_{curr}]^A$, we also provide an order value $[ord]^A$ as input. The order value represents the order of $[x_{curr}]^A$ within the postset $|N.x^\bullet|$. Using $[ord]^A$ as a flag, we assure the look up function LOOKUP_N returns information related to a single node x_k in every iteration. To preserve privacy during look up operation, we store the tables IND, P^{LUT}, A^{LUT}, and I^{LUT} in secret shared format under arithmetic sharing. The sharing type is converted to Boolean sharing when necessary. LOOKUP_N function performs a secure equality check operation for

Table 4.2: Sample look up tables for the Italian traffic road fine management process in Figure 4.1.

	IND	P^{LUT}	A^{LUT}	I^{LUT}
0	0	1	c	0
1	1	1	p	0
2	1	2	s	0
3	1	8	i1	1
4	2	2	p	0
5	2	3	n	0
6	2	8	i2	1
7	3	3	p	0
8	3	3	a	0
9	3	4	j	0
10	3	5	t	0
11	3	8	d	0
12	3	8	i3	1
13	4	8	i4	1
14	4	3	i5	1
15	5	6	l	0
16	6	7	r	0
17	6	8	i6	1
18	7	3	o	0

each index of IND table. If the result of equality is $[1]^A$ for index k, then it copies the index values $[P_k^{LUT}]^A$, $[A_k^{LUT}]^A$, and $[I_k^{LUT}]^A$ into auxiliary variables $[x_k]^A$, $[a_k]^A$, and $[i_k]^A$. To assure that copying is performed only for the intended index, it performs a secure equality check in line 10 and discards the other indices by assigning the values of $[x_k]^A$, $[a_k]^A$, and $[i_k]^A$ for those indices as $[0]^A$. Then, it aggregates the values into $[x_{next}]^A$, $[a_N]^A$, and $[\text{isInv}]^A$. Before terminating, the function checks whether $[x_{next}]^A$ is the final transition of N, x_{final} and, if so, it sets the value of $[\text{endN}]^A$ to $[1]^A$.

LOOKUP_T works similar to LOOKUP_N. However, additional look up tables are not needed for this function. The secure equality check operation is performed on the secret shared index value $[e_{curr}]^B$ for each index of σ. If the result of equality check is $[1]^A$ for index k, then it copies the activity label of $[\sigma_k]^A$. Finally, it checks whether $[e_{curr}]^A$ is the last index of σ, and updates $[\text{endT}]^A$ accordingly.

In Algorithm 11, we provide the pseudocode for the move functions MOVE_ON_MODEL, MOVE_SYNCHRONOUS, and MOVE_ON_LOG operates. Different from the move functions in SCORCH$_{\text{EXH}}$, we store all attributes of the alignment nodes under privacy preservation since their values can leak information about the output. MOVE_ON_MODEL and MOVE_SYNCHRONOUS operate on all elements in the postset of current transition x. For each element, they assign the index values $n_{[i]^A,[j]^A}$, and then call LOOKUP_N function.

Algorithm 10 Look up functions

```
1:  procedure LOOKUP_N([x_curr]^A, [ord]^A)
2:      [x_next] ← [0], [a_N] ← [0], [isInv] ← [0], [endN] ← [0]
3:      [cnt]^A ← [0]
4:      for k = 0 → |IND| do
5:          [eq]^B ← [IND_k]^B ?= [x_curr]^B
6:          [x_k]^A ← [eq]^A · [P_k^LUT]^A
7:          [a_k]^A ← [eq]^A · [A_k^LUT]^A
8:          [i_k]^A ← [eq]^A · [I_k^LUT]^A
9:          if [eq]^B ?= [1]^B then
10:             if [cnt]^B ?≠ [ord]^B then
11:                 [x_k]^A ← [0]^A
12:                 [a_k]^A ← [0]^A
13:                 [i_k]^A ← [0]^A
14:             [cnt]^A ← [cnt]^A + [1]^A
15:         [x_next]^A ← [x_next]^A + [x_k]^A
16:         [a_N]^A ← [a_N]^A + [a_k]^A
17:         [isInv]^A ← [isInv]^A + [i_k]^A
18:     [endN]^B ← [x_next]^B ?= [x_final]^B
19:     return [x_next]^A, [a_N]^A, [isInv]^A, [endN]^A
20:
21: procedure LOOKUP_T([e_curr]^A)
22:     for k = 0 → |σ| do
23:         [eq]^B ← [k]^B ?= [e_curr]^B
24:         [a_k]^A ← [eq]^A · [σ_k]^A
25:         [a_σ]^A ← [a_σ]^A + [a_k]^A
26:     [endT]^B ← [e_curr]^B ?= [|σ| − 1]^B
27:     return [a_σ]^A, [endT]^A
```

The values returned from the look up function are used to assign $n.[\text{endN}]^A$, $n.[x_{curr}]^A$, $n.[c]^A$, and $n.[\gamma]$ for the child nodes n^M and n^S. In MOVE_ON_MODEL, w use $[\text{isInv}]^A$ in the calculation of the alignment cost such that if the transition is invisible then the cost remains the same, otherwise it is increased by 1. LOOKUP_T is not necessary in this function, since the move is performed only on the process model. In MOVE_SYNCHRONOUS, $[\text{isInv}]^A$ is not needed since invisible transitions are not recorded in event log, and the alignment of an invisible transition with an activity label results in an illegal move. Furthermore, in synchronous move function, LOOKUP_T is performed since the move is performed on the log as well. In MOVE_ON_LOG, only LOOKUP_T function is performed since there is no move executed on the process model.

Algorithm 11 $\text{SCORCH}_{\text{PQ}}([\sigma]^A, [N]^A)$ - cont.

procedure MOVE_ON_MODEL(n)
  for $k = 0 \rightarrow |N.x^\bullet|$ **do**
    $n^M_{[i]^A,[j]^A} \leftarrow n_{[i+1]^A,[3j-2]^A}$
    $\{[x_{next}]^A, [a_N]^A, [\text{isInv}]^A, [\text{endN}]^A\} \leftarrow \text{LOOKUP\_N}(n.[x_k]^A, [k]^A)$
    $n^M.[\text{endT}]^A \leftarrow n.[\text{endT}]^A$
    $n^M.[\text{endN}]^A \leftarrow [\text{endN}]^A$
    $n^M.[x_{curr}]^A \leftarrow [x_{next}]^A$
    $n^M.[c]^A \leftarrow [\text{isInv}]^A \cdot n.[c]^A + ([1]^A - [\text{isInv}]^A) \cdot (n.[c]^A + [1]^A)$
    $n^M.[\gamma] \leftarrow n.[\gamma]^A \oplus ([\gg]^A, [a_N]^A)$
    $Q.\text{ENQUEUE}(n^M)$
procedure MOVE_SYNCHRONOUS(n)
  for $k = 0 \rightarrow |N.x^\bullet|$ **do**
    $n^S_{[i]^A,[j]^A} \leftarrow n_{[i+1]^A,[3j-1]^A}$
    $\{[x_{next}]^A, [a_N]^A, [\text{endN}]^A\} \leftarrow \text{LOOKUP\_N}(n.[x_k]^A, [k]^A)$
    $\{[a_\sigma]^A, [\text{endT}]^A\} \leftarrow \text{LOOKUP\_T}([e_{curr}]^A)$
    $n^S.[\text{endT}]^A \leftarrow [\text{endT}]^A$
    $n^S.[\text{endN}]^A \leftarrow [\text{endN}]^A$
    $n^S.[x_{curr}]^A \leftarrow [x_{next}]^A$
    $n^S.[c]^A \leftarrow n.[c]^A + ([a_N]^A - [a_\sigma]^A) \cdot [\text{INT\_MAX}]^A$
    $n^S.[\gamma]^A \leftarrow [n.\gamma]^A \oplus ([a_\sigma]^A, [a_N]^A)$
    $Q.\text{ENQUEUE}(n^S)$
procedure MOVE_ON_LOG(n)
  $n^L_{[i]^A,[j]^A} \leftarrow n_{[i+1]^A,[3j]^A}$
  $\{[a_\sigma]^A, [\text{endT}]^A\} \leftarrow \text{LOOKUP\_T}([e_{curr}]^A)$
  $n^L.[\text{endT}]^A \leftarrow [\text{endT}]^A$
  $n^L.[\text{endN}] \leftarrow n.[\text{endN}]$
  $n^L.[c]^A \leftarrow n.[c]^A + [1]^A$
  $n^L.[\gamma]^A \leftarrow n.[\gamma]^A \oplus ([a_\sigma]^A, [\gg]^A)$
  $Q.\text{ENQUEUE}(n^L)$

4.4. SECURITY ANALYSIS

We design two privacy-preserving protocols for conformance checking under semi-honest security assumption. In this assumption, the parties involved in the computation follows the protocol execution without deviation, but they try to get extra information from the inputs, outputs, and intermediary messages. In our scenario, there are three parties involved in the computations, which are two non-colluding auditors and an auditee. The auditee does not have an active role in computations; rather he provides the secret shared inputs to both auditors. In the second protocol, the auditee can interrupt the protocol to pause execution and checks if an optimal alignment has been computed. When an interruption occurs, the auditee can only access the value of a flag, which indicates whether an optimal alignment has been computed. He cannot access any other

information such as the alignment sequence, the alignment cost, or the index.

The auditors, on the other hand, are responsible for all computations. After receiving the secret shared inputs from the auditee, they collaboratively compute the optimal alignment without colluding. We use a server-aided setting in both our protocols. In this setting, our goal is to hide the content of the input trace, the output alignment (including its length), and the execution order of the places in process model from the auditors. In both protocols, the input and output data are protected and processed using arithmetic circuits and Boolean circuits, which guarantee information-theoretic security against semi-honest adversaries and against malicious adversaries in the existence of honest majority [19, 30, 33]. To hide the execution order, in $\text{SCORCH}_{\text{EXH}}$, we exhaustively compute all possible alignments and find the optimal one among them securely. Since the protocol is exhaustive, it does not leak any information about the execution order.

In $\text{SCORCH}_{\text{PQ}}$, we compute alignments using a priority queue where hiding the execution order of the places is more challenging, since the protocol chooses the alignment node with the minimal cost at every iteration. To guarantee the security of the protocol, we should hide the executed place and the activity name in each iteration. By using secure lookup tables, we achieve our goal of hiding the place and activity information. A second challenge in $\text{SCORCH}_{\text{PQ}}$ is hiding the number of iterations. The advantage of the priority queue is to reduce the number of iterations that enable to compute alignments on larger Petri nets and traces. However, reducing the number of iterations might leak information about the length of the output, which violates our security goal. Thus, to prevent the leakage of the output length, we set a threshold for the number of iterations, which should be greater than the length of the longest possible alignment. This threshold guarantees that the number of iterations is adequate to generate the longest possible trace in the best case. Despite using a threshold reduces the efficiency of computations, it improves the security of the protocol.

4.5. EXPERIMENTS

To measure the performance of our protocols, we implemented them in C++ using the ABY library. We chose 32-bit secret shares for ABY operations. We evaluated the protocols through two experiments using two real-life datasets which are the Italian road fine management process dataset [27] and a credit requirement (CR) process of a bank [26]. In the first experiment, we compare the two protocols with respect to their computation and communication cost. In the second experiment, we measure the scalability of $\text{SCORCH}_{\text{PQ}}$. We performed our experiments on a machine running Ubuntu 18.04 LTS with a 64-bit microprocessor and 16 GB of RAM, with Intel Core i7-4770, 3.40 GHz x 8.

4.5.1. EXPERIMENT 1: $\text{SCORCH}_{\text{EXH}}$ VS $\text{SCORCH}_{\text{PQ}}$

In the first experiment, we compared the performance of the two proposed protocols with respect to their computation and communication cost. The dataset we used in the first experiment is the credit requirement event log of a bank [26].

Due to the limitations of $\text{SCORCH}_{\text{EXH}}$ protocol, the Petri net and the event log used in the experiment should have certain restrictions. Specifically, the Petri net should not

Table 4.3: Comparison of the performance of $\text{SCORCH}_{\text{EXH}}$ and $\text{SCORCH}_{\text{PQ}}$ on CR event logs for a single optimal alignment computation.

	$\text{SCORCH}_{\text{EXH}}$	$\text{SCORCH}_{\text{PQ}}$
#alignment nodes	54271	57
Computation (ms)	21528.90	558.38
Communication (MB)	752.10	21.66

contain any loops and the number of transitions and the length of trace should not be too large. Thus, we modified the CR dataset by reducing the number of activities in the Petri net and the event log, and removing the loops. The modified event log contains 10035 traces, each of which contains approximately 8 events. Figure 4.5 illustrates the process model for CR process. It is worth noting that, even if the log is relative small, we encountered performance issues when applying the $\text{SCORCH}_{\text{EXH}}$ protocol. Therefore, to be able to compare the two protocols, we filtered out the first two activities from the Petri net and from the log, resulting into traces consisting of 6 events.

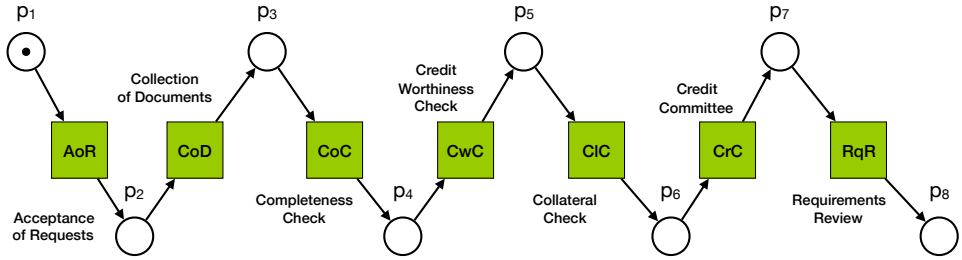


Figure 4.5: The credit requirement process of a bank.

We executed $\text{SCORCH}_{\text{EXH}}$ and $\text{SCORCH}_{\text{PQ}}$ to compare their performance on memory usage, computation time, and communication cost. We measured the average computation time in milliseconds and the bandwidth usage in MB for the computation of a single alignment. To measure the memory usage, we computed the average number of alignment nodes generated in the computation of a single alignment. Table 4.3 summarizes the results of comparison.

The results show that $\text{SCORCH}_{\text{PQ}}$ outperforms $\text{SCORCH}_{\text{EXH}}$ in computation, communication and memory cost. Finding an optimal alignment among all possible alignment sequences requires to generate 54271 alignment nodes. However, using a private priority queue, we can reduce the number of nodes by almost a factor of 1000. Regarding the computation cost, $\text{SCORCH}_{\text{PQ}}$ approximately 38 times more efficient than $\text{SCORCH}_{\text{EXH}}$ in the computation of the optimal alignments. Finally, in communication cost, $\text{SCORCH}_{\text{PQ}}$ needs 35 times less bandwidth usage. The exhaustive nature of $\text{SCORCH}_{\text{EXH}}$ degrades the feasibility of the protocol despite its complete security guarantees.

4.5.2. EXPERIMENT 2: MEASURING THE SCALABILITY OF SCORCH$_{PQ}$

In the second experiment, we measure the scalability of SCORCH$_{PQ}$ to assess the feasibility of our protocol on larger datasets. We performed the experiment on the event logs of the Italian road traffic fine management dataset [27]. The Petri net corresponding to the event log is shown in Figure 4.1. As it can be observed from the figure, the net contains loops, which can only be handled with SCORCH$_{PQ}$. It contains 9 places with 11 unique activity labels and 6 invisible transitions. The event log of the road traffic fine management process contains a total of 1122940 events on 150370 traces. The number of the events in a single trace ranges between 2 and 20. We set the threshold value for for the number of iterations as 20 (See line 8 in Algorithm 9).

We measured the computation time and the bandwidth usage to observe scalability of our protocol. The plots in Figure 4.6 and in Figure 4.7 show the computational and communication performance with respect to the size of the optimal alignment, respectively. As shown in the plots, the change in the computation and communication cost is not always directly proportional to the increase in the optimal alignment size. Indeed, for larger alignment sequences, such as 18 or 19, the computations are almost 5 times faster than the alignments of half of their length. This situation holds for the bandwidth usage, as well. Thus, to better understand the change in the performance of SCORCH$_{PQ}$, we analyzed the average number of alignment nodes generated to compute the alignment. The bar charts in Figure 4.6 and Figure 4.7 show the average number of alignment nodes generated for all possible alignment sizes. The results show that the performance of SCORCH$_{PQ}$ is directly affected by the number of generated alignment nodes, rather than the size of the alignments. Our experiment on the road traffic fine management log show that in a worst case performance, which requires the generation of 521 alignment nodes, SCORCH$_{PQ}$ can compute an optimal alignment approximately in 22 seconds with 980 MB bandwidth usage.

Our experiments show that our protocols can compute optimal alignments with realistic computation and communication cost while guaranteeing privacy preservation. While the performance of SCORCH$_{EXH}$ is limited to smaller datasets without loops, our second protocol SCORCH$_{PQ}$ can scale well for larger datasets and can handle loops in the process.

4.6. RELATED WORK

In this paper, we propose two protocols to compute conformance checking under privacy preservation. Our research is motivated by the need of assuring the confidentiality of audit logs, which is a well-suited application for secure conformance checking. To the best of our knowledge, our proposal is the first attempt in privacy-preserving conformance checking. However, there exist several proposals for privacy preservation in auditing. Below we discuss the relevance of our proposal with respect to the existing work in privacy-preserving auditing.

A solution for privacy-preserving auditing, which is closely related to our protocols, is proposed by Guanciale et al. in [36]. The authors deal with the problem of log auditing and process fusion in business process engineering using secure multiparty computation (MPC). Rather than focusing on internal auditing, they use log auditing in the comparison of the logs of business partners to detect failures, inefficiencies, and errors in the

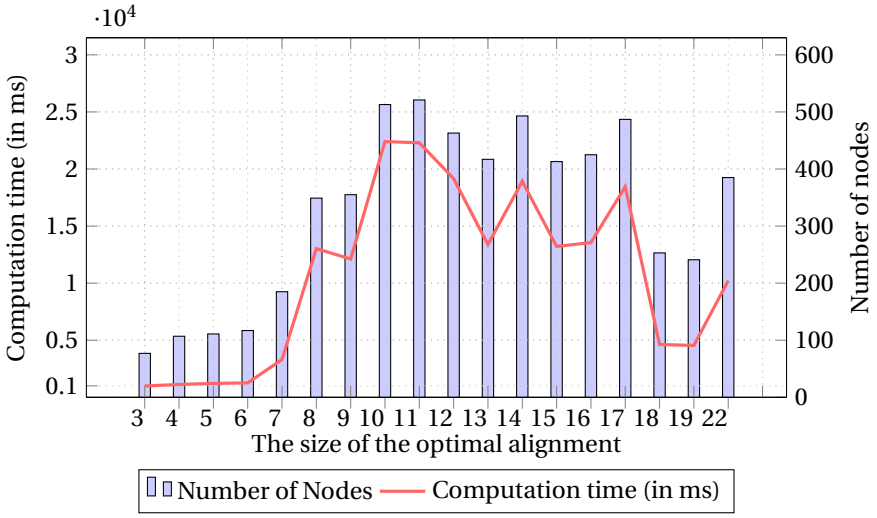


Figure 4.6: Computation cost of SCORCH$_{PQ}$ with respect to the size of the optimal alignment and the number of nodes generated.

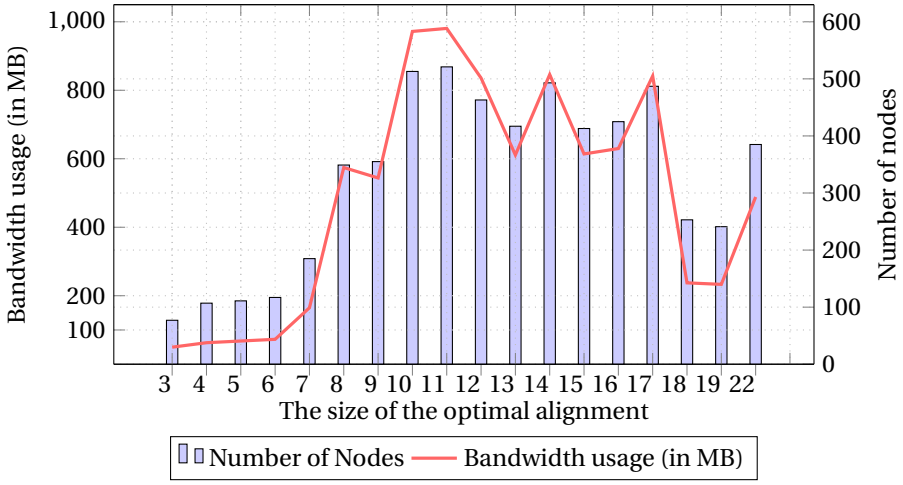


Figure 4.7: Bandwidth usage of SCORCH$_{PQ}$ with respect to the size of the optimal alignment and the number of nodes generated.

process. Different from our protocol, the MPC technique used in [36] is performed by three parties using the universally composable arithmetic black box model [37]. Their proposal first transforms a process model to a finite state automata and then checks whether the alphabet in the log matches the alphabet of the finite state machine. Despite its efficiency, the solution proposed in [36] is not applicable for conformance checking since it does not measure the cost of misalignment. Rather the proposed solutions returns a binary value that indicates a match or mismatch between the log and the process.

A common research field in secure auditing is assuring forward security of audit logs, i.e. preventing the modification of the logs by a malicious attacker [38–42]. Rather than confidentiality, the main goal of these works is to assure integrity and authentication in logging through the use of cryptographic hashes and signatures [38–40, 42]. If the confidentiality of logs is required, the logs are encrypted using identity based encryption (IBE) schemes [39, 41]. Using IBE schemes, efficient keyword search operation can be performed on encrypted logs. Despite their efficiency in search, the aforementioned IBE schemes are not suitable for the type of analysis supported by our protocols since these schemes do not usually support homomorphic operations that are necessary for secure computation on protected data. There exists several homomorphic cryptosystems that also support IBE such as [43, 44]; however, these schemes are not practical for the implementation of our protocols.

4.7. CONCLUSION

Conformance checking is a promising technique for auditing with its comprehensive, automated solutions. However, existing conformance checking mechanisms do not provide privacy protection of logs. In this paper, we proposed two protocols for secure conformance checking, SCORCH$_{\text{EXH}}$ and SCORCH$_{\text{PQ}}$. Both protocols aim to hide the input, output and execution order in the computation of optimal alignments. The output of the conformance checking is revealed only to an authorized party. To the best of our knowledge, we present the first protocols for secure conformance checking using provably secure cryptographic primitives.

Our first protocol, SCORCH$_{\text{EXH}}$, guarantees privacy preservation with an exhaustive computation strategy. Our second protocol SCORCH$_{\text{PQ}}$, on the other hand, uses a private priority queue to reduce the number of alignments to be computed to find an optimal alignment. However, eliminating the need to compute all possible alignments might leak some information about the output of SCORCH$_{\text{PQ}}$. To minimize the information leakage, we set a lower limit for the number of alignments to be computed. The performance evaluation of our protocols on real datasets shows promising results for the application of privacy preservation in conformance checking. SCORCH$_{\text{PQ}}$ can scale well with the growing data size and can handle complex processes with loops.

The current design of our protocols requires some information about the process model and the event log, such as the number of transitions in the Petri net and the number of events in a trace, in plaintext. This information can be considered public, and its leakage is not a severe risk for security. However, in the future, we aim to extend our work by protecting the number of transitions and traces to improve the security of our protocols.

REFERENCES

[1] W. M. P. van der Aalst, K. M. van Hee, J. M. E. M. van der Werf, and M. Verdonk, *Auditing 2.0: Using process mining to support tomorrow's auditor,* IEEE Computer **43**, 90 (2010).

[2] J. McCafferty, *Google's pay gap internal audit yields surprising result,* (2019).

[3] J. McCafferty, *Survey: Companies still struggle on SOX compliance,* (2019).

[4] W. M. P. van der Aalst, *Process Mining - Data Science in Action, Second Edition* (Springer, 2016).

[5] W. M. P. van der Aalst, A. Adriansyah, and B. F. van Dongen, *Replaying history on process models for conformance checking and performance analysis,* Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery **2**, 182 (2012).

[6] D. Barr-Pulliam, *Engaging Third Parties for Internal Audit Activities*, Tech. Rep. (The Global Internal Audit Common Body of Knowledge, 2015).

[7] S. Bergquist and S. Elofsson, *The collaboration between auditors and IT-auditors: The effects on the audit profession*, Tech. Rep. (Uppsala University, 2016).

[8] J. McCafferty, *Companies increasingly outsourcing internal audit,* (2018).

[9] C. Dwork, *Differential privacy,* in *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II* (2006) pp. 1–12.

[10] N. Li, W. H. Qardaji, D. Su, and J. Cao, *Privbasis: Frequent itemset mining with differential privacy,* PVLDB **5**, 1340 (2012).

[11] E. Shen and T. Yu, *Mining frequent graph patterns with differential privacy,* in *International Conference on Knowledge Discovery and Data Mining* (ACM, 2013) pp. 545–553.

[12] G. Jagannathan, K. Pillaipakkamnatt, and R. N. Wright, *A practical differentially private random decision tree classifier,* Trans. Data Privacy **5**, 273 (2012).

[13] M. Abadi, A. Chu, I. J. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, *Deep learning with differential privacy,* in *Conference on Computer and Communications Security* (ACM, 2016) pp. 308–318.

[14] C. Fontaine and F. Galand, *A survey of homomorphic encryption for nonspecialists,* EURASIP J. Information Security **2007** (2007).

[15] M. Beye, Z. Erkin, and R. L. Lagendijk, *Efficient privacy preserving k-means clustering in a three-party setting,* in *International Workshop on Information Forensics and Security* (IEEE, 2011) pp. 1–6.

[16] J. W. Bos, K. E. Lauter, and M. Naehrig, *Private predictive analysis on encrypted medical data,* Journal of Biomedical Informatics **50**, 234 (2014).

[17] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser, *Machine learning classification over encrypted data,* in *Annual Network and Distributed System Security Symposium* (2015).

[18] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. E. Lauter, M. Naehrig, and J. Wernsing, *Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy,* in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016* (JMLR.org, 2016) pp. 201–210.

[19] O. Goldreich, S. Micali, and A. Wigderson, *How to play any mental game or A completeness theorem for protocols with honest majority,* in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA* (ACM, 1987) pp. 218–229.

[20] J. Liu, M. Juuti, Y. Lu, and N. Asokan, *Oblivious neural network predictions via minionn transformations,* in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017* (ACM, 2017) pp. 619–631.

[21] P. Mohassel and P. Rindal, *Aby^3: A mixed protocol framework for machine learning,* in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018* (ACM, 2018) pp. 35–52.

[22] P. Mohassel and Y. Zhang, *Secureml: A system for scalable privacy-preserving machine learning,* in *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017* (IEEE Computer Society, 2017) pp. 19–38.

[23] S. Kamara, P. Mohassel, and M. Raykova, *Outsourcing multi-party computation,* IACR Cryptology ePrint Archive **2011**, 272 (2011).

[24] V. Nikolaenko, S. Ioannidis, U. Weinsberg, M. Joye, N. Taft, and D. Boneh, *Privacy-preserving matrix factorization,* in *Conference on Computer and Communications Security* (ACM, 2013) pp. 801–812.

[25] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft, *Privacy-preserving ridge regression on hundreds of millions of records,* in *Symposium on Security and Privacy* (IEEE, 2013) pp. 334–348.

[26] A. Djedovic, A. Karabegovic, Z. Avdagic, and S. Omanovic, *Innovative approach in modeling business processes with a focus on improving the allocation of human resources,* Mathematical Problems in Engineering **2018** (2018).

[27] F. Mannhardt, M. de Leoni, H. A. Reijers, and W. M. P. van der Aalst, *Balanced multi-perspective checking of process conformance,* Computing **98**, 407 (2016).

[28] T. Schneider, *Engineering Secure Two-Party Computation Protocols - Design, Optimization, and Applications of Efficient Secure Function Evaluation* (Springer, 2012).

[29] A. C. Yao, *Protocols for secure computations (extended abstract),* in *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982* (IEEE Computer Society, 1982) pp. 160–164.

[30] D. Demmler, T. Schneider, and M. Zohner, *ABY - A framework for efficient mixed-protocol secure two-party computation,* in *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015* (The Internet Society, 2015) pp. 1–15.

[31] W. Henecka, S. Kögl, A. Sadeghi, T. Schneider, and I. Wehrenberg, *TASTY: tool for automating secure two-party computations,* in *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010* (ACM, 2010) pp. 451–462.

[32] D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella, *Fairplay - secure two-party computation system,* in *Proceedings of USENIX Security Symposium* (2004) pp. 287–302.

[33] D. Beaver, *Efficient multiparty protocols using circuit randomization,* in *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings* (Springer, 1991) pp. 420–432.

[34] E. Kushilevitz and R. Ostrovsky, *Replication is NOT needed: SINGLE database, computationally-private information retrieval,* in *Annual Symposium on Foundations of Computer Science* (1997) pp. 364–373.

[35] O. Goldreich, *Towards a Theory of Software Protection and Simulation by Oblivious RAMs,* in *Symposium on Theory of Computing* (ACM, 1987) pp. 182–194.

[36] R. Guanciale, D. Gurov, and P. Laud, *Business process engineering and secure multi-party computation,* Cryptology and Information Security Series **13**, 129 (2015).

[37] I. Damgård and J. B. Nielsen, *Universally composable efficient multiparty computation from threshold homomorphic encryption,* in *Advances in Cryptology* (2003) pp. 247–264.

[38] J. E. Holt, *Logcrypt: forward security and public verification for secure audit logs,* in *Australasian Symposium on Grid Computing and e-Research and Australasian Information Security Workshop* (2006) pp. 203–211.

[39] S. E. Oh, J. Y. Chun, L. Jia, D. Garg, C. A. Gunter, and A. Datta, *Privacy-preserving audit for broker-based health information exchange,* in *Conference on Data and Application Security and Privacy* (ACM, 2014) pp. 313–320.

[40] M. A. Shah, R. Swaminathan, and M. Baker, *Privacy-preserving audit and extraction of digital contents,* IACR Cryptology ePrint Archive **2008**, 186 (2008).

[41] B. R. Waters, D. Balfanz, G. Durfee, and D. K. Smetters, *Building an encrypted and searchable audit log,* in *Network and Distributed System Security Symposium* (2004).

[42] A. A. Yavuz and P. Ning, *BAF: an efficient publicly verifiable secure audit logging scheme for distributed systems,* in *Annual Computer Security Applications Conference* (ACM, 2009) pp. 219–228.

[43] M. Clear, A. Hughes, and H. Tewari, *Homomorphic encryption with access policies: Characterization and new constructions,* in *Progress in Cryptology - AFRICACRYPT 2013, 6th International Conference on Cryptology in Africa, Cairo, Egypt, June 22-24, 2013. Proceedings* (2013) pp. 61–87.

[44] C. Gentry, A. Sahai, and B. Waters, *Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based,* in *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I* (2013) pp. 75–92.

5

PRIVATE NEURAL NETWORK PREDICTIONS

The rise of cloud computing technology led to a paradigm shift in technological services that enabled enterprises to delegate their data analytics tasks to cloud servers which have domain-specific expertise and computational resources for the required analytics. Machine Learning as a Service (MLaaS) is one such service which provides the enterprises to perform machine learning tasks on a cloud platform. Despite the advantage of eliminating the need for computational resources and domain expertise, sharing potentially sensitive data with the cloud brings a privacy risk to the enterprises. In this chapter, we propose SwaNN, a protocol to perform neural network predictions for MLaaS under privacy preservation. SwaNN brings together two well-known techniques for secure computation: partially homomorphic encryption (PHE) and secure two-party computation (2PC), and computes neural network predictions by switching between the two methods. The hybrid nature of SwaNN enables to maintain the accuracy of predictions and to optimize the computation time and bandwidth usage. Our experiments show that SwaNN achieves a good balance between computation and communication cost in neural network predictions compared to the state-of-the-art proposals.

This chapter is based on the paper "*SwaNN: Switching among Cryptographic Tools for Privacy-Preserving Neural Network Predictions*" by G. Tillem, B. Bozdemir, and M. Önen (under review).

5.1. INTRODUCTION

Neural networks are a method of supervised machine learning which aims to solve a classification problem. It computes the classification in two phases: a training phase in which a model is trained from previous observations whose classifications are known beforehand; a prediction phase in which a classification is computed for a new observation using the trained model [1].

Although the research on neural networks dates back to 1980s [2], they had not been commonly used due to their long training times. With the recent technological advances and the adaptation of GPUs in computation systems, the training time for neural networks is reduced significantly [3]. The improvement in performance triggered the popularity of neural networks, which in turn provided an outstanding success in certain fields such as image classification [3, 4], face recognition [5], and board games [6].

The success of neural networks attracted many companies to apply it to their businesses. However, it is difficult for companies to successfully benefit from neural networks without having adequate computational resources and expertise in machine learning. Machine Learning as a Service (MLaaS) emerged as a solution to this problem. MLaaS enables the clients to outsource their machine learning tasks to a cloud platform which has computational resources and machine learning expertise [7]. A major risk that is challenging enterprises in using MLaaS is the sensitivity of the data sent to the cloud. The concern of exposing privacy-sensitive data in MLaaS services requires the design of privacy-preserving protocols for machine learning methods.

In this paper, we aim to design one such protocol for MLaaS to compute neural network predictions under privacy preservation. We assume that the network model has already been computed during a previous training phase, and we only focus on the privacy of data items during the prediction phase. Indeed, this problem drew the attention of researchers recently and several mechanisms that provide privacy protection in neural network predictions are proposed. The existing solutions that rely on cryptographic tools can be regrouped mainly in two categories. The solutions that are based on homomorphic encryption (HE) [1, 8–10] enable computation of linear operations and low-degree polynomials non-interactively, where computations are performed by an external semi-trusted server. These solutions usually incur high computational cost due to the expensive nature of homomorphic encryption systems. Also, the restriction of linear and low-degree polynomial operations degrades the accuracy of prediction. Secure two-party computation (2PC)-based solutions [11–13], on the other hand, provide more realistic computation performance and seem better in maintaining the accuracy of predictions. However, the interactive nature of 2PC-based solutions leads to a higher bandwidth usage compared to HE-based alternatives.

Having studied existing solutions, we aim to take the best of both worlds and optimize the computational and the communication overhead at the same time. We propose a hybrid protocol, **SwaNN**, which switches the computations between HE and 2PC. Instead of using leveled or somewhat homomorphic encryption, we make use of partially homomorphic encryption (more specifically the additively homomorphic Paillier encryption) to perform linear operations over encrypted data. This also helps the solution reduce the computational cost. Non-linear operations are supported thanks to the use of 2PC. We show how to easily switch from one cryptographic tool to the other.

The combination of these two cryptographic tools helps maintain the accuracy of predictions. The idea of using a hybrid protocol for private neural network predictions is proposed in Gazelle [14] as well, which combines Yao?s garbled circuits with a dedicated lattice-based additively homomorphic encryption scheme. Our proposal differs from Gazelle by using well-known simple cryptographic tools, which make the adoption of our proposal more practical.

SwaNN is designed to support two different settings: a client-server setting and a non-colluding two-server setting. In the client-server setting, the majority of operations are delegated to the server, and the client helps the server in intermediate steps. In the two-server setting, the servers are provided the data beforehand, and they perform the computations simultaneously, with a balanced workload on both servers. Our contributions can be summarized as follows:

- We propose a hybrid protocol for neural network predictions, which is based on the additively homomorphic Paillier encryption scheme and secure two-party computation. We show how each underlying operation can be supported easily with the use of these two schemes only.
- Our protocol is flexible since it is suitable both for the client-server setting and the non-colluding two-server setting.
- Compared to existing works, our protocol proposes several optimizations for the computations in the linear layers of neural networks which improves the efficiency in terms of computation cost. These optimisations consist of some data packing dedicated to the Paillier cryptosystem and the use of multi-exponentiation algorithm to reduce the cost of multiplications.
- The empirical results show that our protocol can compute the prediction on a neural network with two activation layers in 10 seconds with 1.73 MB bandwidth usage which is 30-fold better in computation cost than the state-of-the-art HE-based solution and 27-fold more efficient in bandwidth usage than the state-of-the-art 2PC-based solution.

Section 5.2 introduces neural networks and the underlying cryptographic tools used in SwaNN. In Section 5.3, we discuss the contribution of our work along with the prior work. We describe our protocol in Section 5.4. In Section 5.5, we present the empirical evaluation of our work. We conclude our paper in Section 5.6.

5.2. PRELIMINARIES

In this section, we present the necessary background information on convolutional neural networks and the cryptographic primitives we use in our design. Table 5.1 summarizes the notation we use throughout the paper.

5.2.1. CONVOLUTIONAL NEURAL NETWORKS

Convolutional neural networks are specifically designed for image recognition. They combine a series of layers to perform classification. Each layer takes an input $\mathbf{X}$, evaluates a function f on the input along with a weight matrix $\mathbf{W}$, and returns an output $\mathbf{Y}$ to

Table 5.1: Notation table.

Symbol	Explanation
$\mathbf{X} = \{x_{0,0}, x_{0,1}, \ldots\}$	Input matrix
$\mathbf{Y} = \{y_{0,0}, y_{0,1}, \ldots\}$	Output matrix
$\mathbf{W} = \{w_{0,0}, w_{0,1}, \ldots\}$	Model weight matrix
$\mathbf{B} = \{b_{0,0}, b_{0,1}, \ldots\}$	Bias matrix
$f(\mathbf{X}, \mathbf{W}) = \mathbf{Y}$	Function f that operates on inputs $\mathbf{X}, \mathbf{W}$ and returns output $\mathbf{Y}$
N	Plaintext modulus of the Paillier cryptosystem
N^2	Ciphertext modulus of the Paillier cryptosystem
ℓ	Bit size for 2PC operations
κ	Security parameter
$a \in_R A$	a is chosen uniformly randomly from A
$\odot$	Dot product symbol
$\otimes$	Matrix multiplication symbol
$[\cdot]$	Paillier ciphertext
$\langle \cdot \rangle_i$	Input share of party i for 2PC operations

the subsequent layer. Figure 5.1 illustrates an overview of the neural network structure. The first layer of the neural network is the input layer, where the input is provided to the network. The last layer is the output layer, where the result of the classification is revealed. The layers between input and output are called hidden layers. Each hidden layer evaluates the function associated with that layer on its input and delivers the output to the next layer.

Below we describe the most common hidden layers used in convolutional neural networks.

Convolutional Layer (`Conv`) is based on convolutional filtering in image processing. It applies a filter, $\mathbf{W}$, on the input $\mathbf{X}$ by sliding the filter every time to work on each index of the input matrix. The sliding factor is called stride (s). The size of the input can be adjusted to fit to filter size by appending some border values which is called padding (p). The operation performed in convolutional layer is a dot product such that

$$f(\mathbf{X}, \mathbf{W}) = \mathbf{W} \odot \mathbf{X} + \mathbf{B} \tag{5.1}$$

$$= \sum w_{i,j} \times x_{i,j} + b_{i,j}. \tag{5.2}$$

Figure 5.2 shows the operation of convolutional filter on the input.

Fully Connected Layer (`FC`) connects each neuron in the current layer to each neuron in the previous layer along with a weight value. The operation is a matrix multiplication

$$\mathbf{X}^t = \mathbf{X}^{t-1} \otimes \mathbf{W}, \tag{5.3}$$

where $\mathbf{X}^t$ are the neurons of the current layer and $\mathbf{X}^{t-1}$ are the neurons of the previous layer.

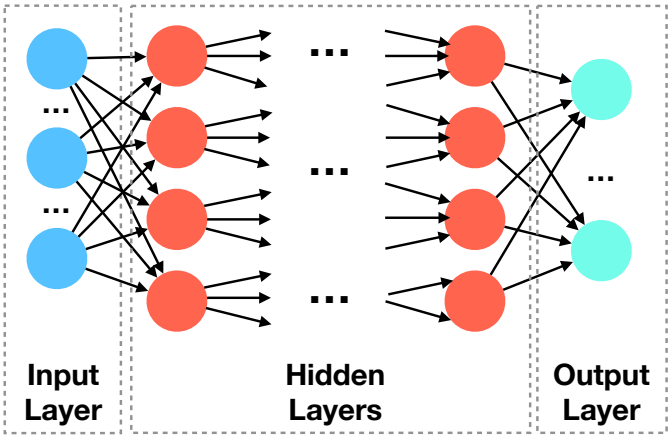


Figure 5.1: An overview of the neural network structure with input, output, and hidden layers.

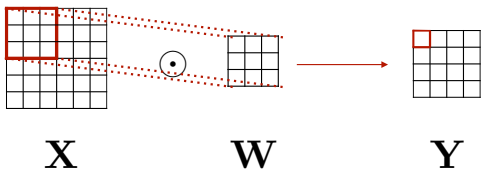


Figure 5.2: Convolutional filtering in convolutional layer.

Pooling Layer (`Pool`) is a scaling layer which reduces the size of the input matrix. Reduction is performed by sliding a filter on the input and performing the pooling operation on each area that is covered by the filter. The two common types of pooling are

- *max pooling* where the maximum value within the area covered by the filter is selected.
- *average pooling* where the average of the values within the area covered by the filter is selected.

Unlike previous layers, the pooling operations are nonlinear.
Activation Layer (`Act`) is also a nonlinear layer. In this layer, a nonlinear activation function $f(x_{i,j}) = y_{i,j}$ is applied to each neuron of the input layer. The size of the output is same with the size of the input. Most commonly used activation functions are

$$\textit{Sigmoid: } \frac{1}{1+e^{-x_{i,j}}}, \tag{5.4}$$

$$\textit{Hyperbolic tangent (tanh): } \frac{e^{2x_{i,j}}-1}{e^{2x_{i,j}}+1}, \tag{5.5}$$

$$\textit{Rectified Linear Units (ReLU): } \max(0, x_{i,j}). \tag{5.6}$$

5.2.2. HOMOMORPHIC ENCRYPTION

Homomorphic property enables a cryptosystem to perform operations on the encrypted input without decryption. If a cryptosystem enables both additions and multiplications under encryption, it is called fully homomorphic whereas if it supports a single type of operation, it is called partially homomorphic. Despite their flexibility on performing both types of operations, fully homomorphic cryptosystems are expensive in computation. In contrast, partially homomorphic schemes are more efficient with their reasonable computation cost.

In this paper, we use a partially homomorphic cryptosystem, namely the Paillier cryptosystem [15] which supports additive homomorphism. The public key of the Paillier cryptosystem is (N, g), where N is the product of two large primes p and q, and $g \in \mathbb{Z}^*_{N^2}$. The private key is (λ, μ), where $\lambda = lcm(p-1, q-1)$ and $\mu = (L(g^\lambda \mod N^2))^{-1} \mod N$. The encryption function of the Paillier cryptosystem is probabilistic such that every encryption of the same plaintext results in a different ciphertext. Encryption of a message $m \in \mathbb{Z}_N$ on modulus N is computed as $E(m) = g^m \cdot r^N \mod N^2$, where $r \in \mathbb{Z}^*_{N^2}$. An encrypted message $E(m)$ can be decrypted using the formula $m = L(c^\lambda \mod N^2) \cdot \mu \mod N$. As it is described in [16], the decryption function of the Paillier cryptosystem supports threshold decryption. In our paper, when necessary, we use a 2-out-of-2 variant of the threshold decryption which distributes the private key among two parties. Successful decryption requires both parties to compute the decryption function.

Using the Paillier cryptosystem, it is possible to compute addition and scalar multiplication on encrypted messages as depicted in the following equations

$$\begin{aligned} E(m_1) \times E(m_2) &= g^{m_1} \cdot r_1^N \times g^{m_2} \cdot r_2^N \mod N^2 \\ &= g^{m_1+m_2} \cdot (r_1 \cdot r_2)^N \mod N^2 \\ &= E(m_1 + m_2), \end{aligned} \tag{5.7}$$

$$\begin{aligned} E(m)^c &= g^{cm} \cdot (r^c)^N \mod N^2 \\ &= E(c \cdot m). \end{aligned} \tag{5.8}$$

We refer readers to [15] for the details of the cryptosystem. In the rest of the paper, we represent a Paillier ciphertext with $[\cdot]$ symbol.

5.2.3. SECURE TWO-PARTY COMPUTATION

Secure two-party computation enables two parties to jointly compute a function f on their inputs without revealing the inputs to each other. In our work, we use two different methods for secure two-party computation which are arithmetic secret sharing [17] and Boolean secret sharing [18].

Arithmetic secret sharing: Given two parties P_1, P_2, their arithmetic shares on an ℓ-bit value m are $\langle m \rangle_1$ and $\langle m \rangle_2$ such that $\langle m \rangle_1 + \langle m \rangle_2 \equiv m \mod 2^\ell$. Arithmetic sharing enables computation of addition and multiplication operations on secretly shared values. The addition of two secret shared values is computed locally by each party as $\langle z \rangle_i = \langle x \rangle_i + \langle y \rangle_i$. Computing multiplication is, however, more complicated. It can be computed using Beaver's multiplication triplet technique [17]. We refer readers to [17] for a detailed explanation of the technique.

Boolean secret sharing: Given the parties P_1, P_2, their Boolean shares on a single bit value m are $\langle m \rangle_1$ and $\langle m \rangle_2$ such that $\langle m \rangle_1 \oplus \langle m \rangle_2 \equiv m \mod 2$, where $\oplus$ is the XOR operation. Boolean circuits can compute both linear and nonlinear operations.

In arithmetic sharing, additions can be computed locally without any additional cost. A multiplication operation requires some additional computation and communication cost; however, it is less expensive than the multiplication in Boolean sharing [19]. Therefore, in our protocol, we use arithmetic sharing for addition and multiplication operations. When other types of operations such as comparisons are needed, we use Boolean sharing.

5.3. PRIOR WORK

In this section, we regroup existing privacy preserving neural networks into several categories based on the underlying privacy enhancing technology. We further highlight their relevance with respect to our protocol.

The first category of solutions consists of solutions based on secure multi-party computation. In [11], SecureML designs a privacy-preserving neural network training and classification method using 2PC, where clients secretly share their own private data among two non-colluding servers. SecureML builds the model with the stochastic gradient descent method. Authors compute ReLU using garbled circuits and implement polynomial approximations of nonlinear functions such as the sigmoid and softmax functions. Additionally, a solution for switching between arithmetic and Yao's sharing is proposed. As an extension to SecureML, authors [12] propose ABY^3 which shares the private inputs between three non-colluding servers. To securely share sensitive data among three servers, the authors redefine arithmetic, Boolean and Yao's sharings of the ABY framework [19]. MiniONN, proposed by Liu et al. [13], also uses 2PC for privacy-preserving neural network operations. Different from [11] and [12], MiniONN focuses on the prediction phase only. The authors propose a 2PC protocol between the client and the cloud. The client and the cloud additively share each of their input and output values for each layer of the neural network. To ensure data privacy, MiniONN defines oblivious transformations for each CNN operation and implements the transformations using the ABY framework. Furthermore, Rouhani et al. [20] propose DeepSecure which is based on Yao's garbled circuits to securely compute the deep learning model. The authors are able to use sigmoid and tanh as activation functions thanks to the optimization of garbled circuits. Another study which uses 2PC is Chameleon by Riazi et al. [21]. Authors propose a protocol that switches among sharing circuits for secure function evaluation, where two parties jointly perform a computation without disclosing their inputs. Chameleon can be considered as an alternative protocol to ABY [19]. TFEncrypted [22] is another framework which enables secure computation in TensorFlow [23] using secret sharing and secure channels between the parties. Moreover, a very recent scheme named SecureNN [24] uses secure three-party computation for the training and classification phases with convolutional neural networks using the MNIST dataset. SecureNN shares the input and output among two parties using 2-out-of-2 arithmetic shares, and the third party joins the protocols during the online computation. In comparison to SecureNN, SwaNN requires interaction with the client for the computation of the square function only, which is less than compared to the interactions of three parties in Se-

cureNN.

In the second category, we analyze fully homomorphic encryption (FHE)-based solutions. To the best of our knowledge, CryptoNets [8] is the first privacy-preserving neural network protocol which is based on FHE. Authors in [8] use the SEAL library [25] to compute convolutional neural network predictions on encrypted images. Similar to CryptoNets, CryptoDL [26], Chabanne et al. [1] and Ibarrondo et al. [27] use FHE for privacy-preserving neural networks. The main difference with CryptoNets is the fact that they approximate nonlinear functions with higher degree polynomials using different techniques such as Taylor series, numerical methods or Chebyshev polynomials. The use of batch normalization is also proposed to obtain some performance gain. The goal of all these solutions is to keep a good level of accuracy while using FHE to protect the input data. Later on, Bourse et al. [28] uses a conversion of a trained neural network to a Discretized Neural Network (DiNN) using an efficient FHE called TFHE [29]. Authors claim that DiNN can be used for deep neural networks with large number of neurons. Similarly, TAPAS [30] also proposes binary neural networks over TFHE-encrypted data. However, TAPAS differs from [28] mainly due to the ability of the server to update the neural network at any time without the need for the data being re-encrypted by the client. More recent works, namely [31] and [32] propose the idea of training neural networks over FHE-encrypted data and classifying encrypted predictions. In their studies, the client supplies the training data in its encrypted form using its own public key, and the server trains this encrypted data to build the encrypted model. This model is further used by its owner to classify a new encrypted input. Because both the training data and the model are encrypted, the server cannot discover any information on both phases. Authors claim to achieve a reasonable performance. Moreover, Faster CryptoNets [33] is a system employing the sparse encodings over the neural network model and data when it remains encrypted under the FHE scheme. The authors propose some efficient polynomial approximations for activation functions. The scheme also includes a training phase that uses differential privacy to protect the data.

In comparison with existing solutions from these two categories, we propose a hybrid protocol that combines 2PC with partially HE. Our goal in SwaNN is to come up with private neural network predictions by making use of more simple cryptographic tools, where the client can obtain the prediction result without disclosing its input to the server, and the privacy of server's neural network against the client is ensured. Therefore, we propose to take advantage of both privacy enhancing technologies and optimize their respective costs (computational and/or communication cost). To reduce the computational cost, FHE is replaced with the additively homomorphic Paillier encryption scheme. This algorithm is used to compute linear operations and the x^2 function. Additionally, we obtain better performance results for computing nonlinear operations thanks to the use of 2PC.

Few early approaches, such as [9] and [10], also use the Paillier encryption scheme and Yao's garbled circuits. Although these solutions seem similar to our proposal, they study very small neural networks and suffer from significant communication overhead due to frequent client-server interactions. Furthermore, authors in [9] and [10] do not provide any performance results of their solution executed over the encrypted data. Additionally, Gazelle [14] is a secure neural network inference scheme implemented under

a dedicated lattice-based additively homomorphic encryption scheme proposed in the paper. This solution also makes use of Yao's garbled circuits to perform ReLU and to reduce the noise in the ciphertext. Unfortunately, this results in linear growth in computation and communication costs, and it also increases the depth of the circuit. Instead, we make use of a secure square protocol to compute the activation function without Yao's garbled circuits.

Lastly, there exist several works, such as [34], [35], and [36], which propose to combine some machine learning techniques, including neural networks, with trusted hardware.

5.4. SWANN

In the Machine Learning as a Service (MLaaS) model, the client has limited computation capabilities or knowledge of machine learning. Thus, he outsources the computations to the server who has expertise in performing machine learning with adequate computation power. In a desirable scenario, the workload on the client side should be minimized. In this paper, we consider two different scenarios both of which aim to minimize the computations at the client side and the overall computation cost while maintaining privacy.

1st Scenario - Client-Server: In this scenario, a client shares a private image with a server. The server, which holds the neural network model, computes the prediction result on the private image. The majority of the computations are performed by the server. The client helps the server perform decryptions and/or circuit evaluations when it is necessary.

2nd Scenario - Two-Server: To reduce the workload on the client side further, we design a two-server setting where two semi-honest non-colluding servers perform the computations together. The client provides the servers their shares on the input and private keys. Thus, the computations on the client side are completely delegated to the servers. In such a setting to fully utilize the capabilities of both servers, one image can be provided to each server such that at one execution they evaluate two images simultaneously.

In both scenarios, we assume a semi-honest security model, where the parties do not collude. In this security model, parties exactly follow the protocol steps. However, they are curious to obtain some information from their output and intermediary messages. In both of our scenarios, the client's goal is to hide the image content and the result of classification from the server. On the other hand, the server does not want to reveal the model parameters used during computations to the client. In the rest, we explain the computation of private neural network predictions for both scenarios individually.

5.4.1. SCENARIO 1: CLIENT - SERVER

In the client-server scenario, the majority of computations are performed by the server, and the client is involved when intermediary decryptions are needed. Figure 5.3 illustrates our first scenario. The client encrypts an image with his public key and sends it to the server who computes the secret prediction result using the neural network parameters. Depending on the operation performed by the server, the client might involve in the computations.

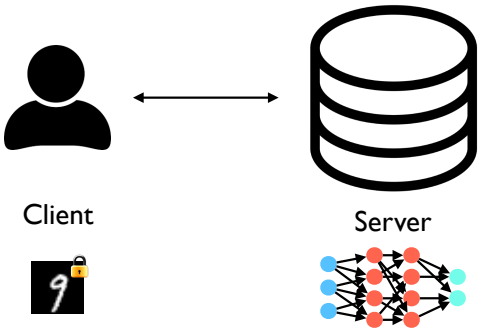


Figure 5.3: Client-server scenario for SwaNN with a single input image.

In Section 5.2.1, we summarize the common layers for convolutional neural networks and the necessary operations to compute the functions in these layers. Below we explain how we can compute these layers under privacy preservation in the client-server scenario. Essentially, we separate the computations into two phases as non-interactive phase and interactive phase. In the non-interactive phase, the operations are performed by the server without the client's involvement. The interactive phase, however, requires the collaboration of the server and the client for computations. By convention, convolutional neural networks start with a convolutional layer. Therefore, we assume that the computations always start with an image that is encrypted under the Paillier cryptosystem by the client. This encrypted image is sent to the server.

NON-INTERACTIVE PHASE

In this phase, the server, who has received the encrypted image, computes the linear layers of the neural network as follows.

Convolutional Layer: The main operation in the convolutional layer is the dot product. Given an input image $\mathbf{X}$ and a weight matrix $\mathbf{W}$, their dot product is computed as $\mathbf{Y} = \sum x_{i,j} \times w_{i,j}$. When the input image is encrypted with the Paillier cryptosystem and the weight matrix is in plaintext, using the homomorphic property of encryption, the dot product is computed as

$$[\mathbf{Y}] = \left[\sum x_{i,j} \times w_{i,j}\right] = \prod \left[x_{i,j}\right]^{w_{i,j}}. \tag{5.9}$$

Since this computation does not require any decryption, it can be performed noninteractively by the server.

Fully Connected Layer: Fully connected layer requires to compute a matrix multiplication. The underlying operation for matrix multiplication is the dot product, but it has to be performed for each column and row pair. Given an encrypted image as input, the fully connected layer is computed by performing Equation 5.9 repetitively.

Mean Pool Layer: Despite the computation of pooling layer is nonlinear, following the convention in the state-of-the-art works [8, 13] we use a linear approximation of the mean pooling operation. Originally, the computation of mean pooling requires the summation of the values within a subgroup and then a division by the subgroup size.

Following the approach in [8, 13], we compute scaled mean pool instead of the mean pool, where the summation is performed, but the division is omitted. The scaled mean pool can be computed by additive homomorphic property of the Paillier cryptosystem without interaction.

INTERACTIVE PHASE

In this phase, the server computes the nonlinear layers of the neural network in collaboration with the client as follows.

Activation Layer: Computing the nonlinear activation function in neural networks is a challenging task when privacy preservation is required. Since the Paillier homomorphic encryption supports only additions, activation functions cannot be computed without performing decryption. In the existing literature on privacy-preserving neural networks, there are two approaches to compute the activation function.

The first approach is to compute a polynomial approximation of the function. CryptoNets [8] and MiniONN [13] use x^2 as the approximation of the sigmoid function. In SwaNN, we propose two solutions to compute the approximation function x^2. Since the Pailler cryptosystem does not support multiplications, as a first solution, we design an interactive secure square function using the additively homomorphic property of the Paillier cryptosystem. Our solution adapts the secure multiplication protocol in [37] to a secure square protocol (see Algorithm 12). Our second solution for the computation of x^2 uses a multiplication operation under arithmetic sharing. The multiplication requires to switch the computations from homomorphic encryption to arithmetic sharing. Later in this section, we explain how we can perform such a switching operation.

Algorithm 12 Secure Square Protocol

Client (pk, sk)		Server (pk)
		$[x], r \in_R \{0,1\}^{\ell+\kappa}$
		$[x_r] \leftarrow [x] \cdot [r]$
		$[x_r] \leftarrow [x + r]$
	$\xleftarrow{[x_r]}$	
$x_r \leftarrow \text{decr}([x_r])$		
$x_r^2 \leftarrow x_r \cdot x_r$		
$[x_r^2] \leftarrow \text{enc}(x_r^2)$		
	$\xrightarrow{[x_r^2]}$	
		$[x_r^2] \cdot \left([r^2] \cdot [x]^{2r}\right)^{-1}$
		$[x^2] \leftarrow \left[x_r^2 - r^2 - 2xr\right]$

The second approach to compute the ReLU activation function using secure two-party computation techniques. MiniONN [13] and SecureML [11] are the state-of-the-art solutions which use arithmetic circuits and Yao's garbled circuits [38] to compute the ReLU activation function. In SwaNN, we adapt a similar approach and use the circuit-based approach when the computation of ReLU is required. We compute ReLU using a comparison gate under Boolean sharing.

Max Pool Layer: Unlike the mean pool layer, we do not use an approximation function for the computation of the max pool layer. Instead, we implement the maximum pooling

using the comparison gates under Boolean sharing. We perform the max pool layer right after the activation layer to reduce the number of switching operations between 2PC and PHE.

Switching between HE and 2PC In the previous subsections, we describe how to compute linear and nonlinear layers of neural networks using partially homomorphic encryption (PHE) and secure two-party computation (2PC). Since linear and nonlinear operations follow each other repetitively, we need a secure switching mechanism between the two cryptographic techniques. We design a protocol for secure switching which is similar to the secure decryption mechanism described in [39]. Algorithm 13 and 14 demonstrate the steps of switching from PHE to 2PC and 2PC to PHE, respectively.

Algorithm 13 PHE to 2PC Secure Switching Protocol

Client (pk, sk)		Server (pk)
		$[x]$, $r \in_R \{0,1\}^{\ell+\kappa}$
		$[x+r] \leftarrow [x] \cdot [r]$
	$\xleftarrow{[x+r]}$	
$x+r \leftarrow \text{decr}([x+r])$		
$x+r \rightarrow \langle x+r\rangle_c + \langle x+r\rangle_s$		
	$\xrightarrow{\langle x+r\rangle_s}$	
		$r \rightarrow \langle r\rangle_c + \langle r\rangle_s$
	$\xleftarrow{\langle r\rangle_c}$	
$\langle x\rangle_c \leftarrow \langle x+r\rangle_c - \langle r\rangle_c$		$\langle x\rangle_s \leftarrow \langle x+r\rangle_s - \langle r\rangle_s$

Switching from PHE to 2PC (Algorithm 13) requires to perform a secure decryption by masking the encrypted value with a random r. Once the client securely decrypts the masked value $x+r$, he creates the secret shares of it for himself and for the server as $\langle x+r\rangle_c$ and $\langle x+r\rangle_s$. In the mean time, the server creates the secret shares of the random r as $\langle r\rangle_c$ and $\langle r\rangle_s$ to remove the mask from the original value x. Finally, both parties perform a local subtraction on their shares $\langle x+r\rangle$ and $\langle r\rangle$ to compute the secret shared value $\langle x\rangle$ which is going to be used in 2PC computations.

Algorithm 14 2PC to PHE Secure Switching Protocol

Client (pk, sk)		Server (pk)
$\langle x\rangle_c$		$\langle x\rangle_s$, $r' \in_R \{0,1\}^{\ell+\kappa}$
		$r' \rightarrow \langle r'\rangle_c + \langle r'\rangle_s$
	$\xleftarrow{\langle r'\rangle_c}$	
$\langle x+r'\rangle_c \leftarrow \langle x\rangle_c + \langle r'\rangle_c$		
		$\langle x+r'\rangle_s \leftarrow \langle x\rangle_s + \langle r'\rangle_s$
	$\xleftarrow{\langle x+r'\rangle_s}$	
$x+r' \leftarrow \langle x+r'\rangle_c + \langle x+r'\rangle_s$		
$[x+r'] \leftarrow enc(x+r')$		
	$\xrightarrow{[x+r']}$	
		$[x] \leftarrow [x+r'] \cdot [r']^{-1}$

Switching from 2PC to PHE (Algorithm 14) reverses the former procedure. It starts with a secret shared value $\langle x\rangle$. Similar to the previous protocol, to prevent the leakage

of the original value the parties reveal it after masking. Thus, the server generates a random mask r' and sends a secret share of the random $\langle r' \rangle_c$ to the client. Both parties perform an addition operation to mask $\langle x \rangle$, and then the server sends the masked value $\langle x + r' \rangle_s$ to the client. Client reveals $x + r'$ by adding the two shares and encrypts it with his public key. In the final step, the server removes the random mask from $[x + r']$ with a homomorphic subtraction.

5.4.2. Scenario 2: Two-Server

The client-server scenario necessitates a certain level of computation power from the client, despite the majority of the operations are performed by the server. To reduce the workload from the client's side, we design a second scenario which outsources the computations to two non-colluding servers. In this scenario, the client provides the input to both servers, and the servers perform the operations and return the result to the client. However, if only a single image is provided to the servers one of the servers is going to be idle during the non-interactive phase of the computations. Thus, we propose to provide one different image to each server to fully utilize the computation capabilities of the servers and classify two images at once.

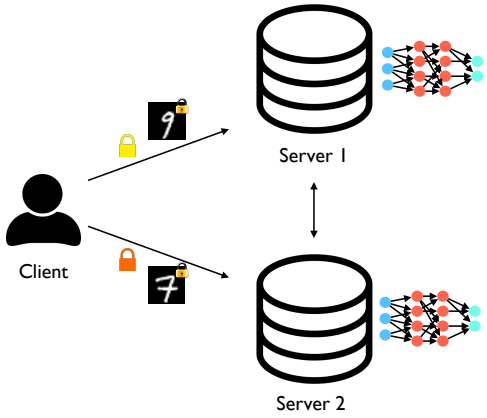


Figure 5.4: Two-server scenario for SwaNN with two input images.

Figure 5.4 illustrates our scenario. The client encrypts two images with his public key and provides one image to each server. Furthermore, he creates shares of the private key for each server as described in [16] and sends the shares to each server. Similar to the first scenario, we divide the computations into two phases as non-interactive and interactive phases. In the non-interactive phase, the servers compute the linear operations on their inputs as the same way described in Section 5.4.1. The interactive phase and the switching phase are also similar to the description in Section 5.4.1, but they differ in the decryption procedure. In the first scenario, the client is responsible for performing the decryption operations. However, in the two-server scenario, the decryption task is also delegated to the servers along with their shares on the secret key. Therefore, the decryption function decr($[\cdot]$) in Algorithm 12 and Algorithm 13 is performed by both servers. To clarify the procedure, in Protocol 15 we illustrate how secure square protocol works

when the computations are delegated to the two servers.

Algorithm 15 Secure Square Protocol in the two-server scenario

Server 1 (pk, sk_1)		Server 2 (pk, sk_2)
		$[x], r \in_R \{0,1\}^{\ell+\kappa}$
		$[x_r] \leftarrow [x] \cdot [r]$
		$[x_r] \leftarrow [x + r]$
		$[x_r]' \leftarrow \text{decr}_2([x_r])$
	$\xleftarrow{[x_r]'}$	
$x_r \leftarrow \text{decr}_1([x_r]')$		
$x_r^2 \leftarrow x_r \cdot x_r$		
$[x_r^2] \leftarrow \text{enc}(x_r^2)$		
	$\xrightarrow{[x_r^2]}$	
		$[x_r^2] \cdot \left([r^2] \cdot [x]^{2r}\right)^{-1}$
		$[x^2] \leftarrow \left[x_r^2 - r^2 - 2xr\right]$

5

While the execution of non-interactive phase can be done by each server locally, the interactive phase requires the involvement of both parties. The servers can execute this phase sequentially based on a predetermined order, or they can execute it in parallel which improves the computation cost further.

5.4.3. SECURITY ANALYSIS

SwaNN aims to compute neural network predictions under the privacy preservation assumption in the semi-honest adversarial model. We assume the semi-honest adversary is non-adaptive and computationally bounded. In this security model, for both of the scenarios we propose, the two parties should not be able to retrieve any additional information from the protocol execution apart from their inputs, outputs, and intermediary messages. We achieve our security goal thanks to the security of the cryptographic techniques we use in the design of SwaNN. Both the Paillier cryptosystem and secure two-party computation are proven to be secure. In the non-interactive phase of SwaNN, the security is guaranteed by the semantic security of the Paillier cryptosystem. The Paillier cryptosystem satisfies semantic security against chosen plaintext attacks under decisional composite residuosity assumption [15]. Thus, in the computation of convolutional, fully connected, and mean pool layers, the server(s) cannot reveal any valuable information from the encrypted messages on the condition that the encryption is performed with a key that meets the current security requirements.

The activation and max pool layers, on the other hand, requires interactive protocols between two parties during which the computations might be switched from homomorphic encryption to secure two-party computation, and vice-versa. Besides the security of the Paillier cryptosystem, arithmetic secret sharing and Boolean secret sharing, which are used in the interactive phase of SwaNN as secure two-party computation techniques, achieve indistinguishability given that the shares are generated from a uniformly random distribution [18]. Assuming that the Paillier cryptosystem and secure two-party computation are secure, the security of the interactive phase of SwaNN can be deduced to the security of switching or decryption operations. In the rest of this section,

we provide a formal security proof using the simulation paradigm [40] to show that the switching and decryption operations can be performed securely. Due to limited space, we provide the proof only for Algorithm 13, which switches the operations from homomorphic encryption to secure two-party computation in the client-server scenario.

In the simulation paradigm, the ideal security setting is outsourcing inputs of both parties to a trusted third party who can perform the computations and return the output. In the real-world setting, the security goal is to show that if an adversary $\mathscr{A}$ can attack the protocol in the real world, then the attack can be also performed by an adversary $\mathscr{S}$ in the ideal world. Since the attacks of $\mathscr{S}$ are not successful in the ideal setting, the attacks in the real world also fail and the protocol is proved to be secure in the real world. In Definition 5.4.1 and Definition 5.4.2, we provide the formal definitions for security and indistinguishability from [40].

Definition 5.4.1 (Computational Indistinguishability)**.** Let $X(a,\kappa)$ and $Y(a,\kappa)$ are two probability ensembles where $a \in \{0,1\}^*$ is the input of the parties and κ is the security parameter. $X(a,\kappa)$ and $Y(a,\kappa)$ are *computationally indistinguishable* (i.e. $X(a,\kappa) \equiv Y(a,\kappa)$) if there exists a negligible function $\mu(\kappa)$ for every nonuniform polynomial time algorithm D, and for every $a \in \{0,1\}^*$ and $\kappa \in \mathscr{K}$ such that

$$|\Pr[D(X(a,\kappa)) = 1] - \Pr[D(Y(a,\kappa)) = 1]| \leq \mu(\kappa). \tag{5.10}$$

Definition 5.4.2 (Definition of Security)**.** P_1 and P_2 are two parties who want to run a protocol π on their inputs x and y to compute a functionality $f(x,y)$ which outputs $f_1(x,y)$ and $f_2(x,y)$ for each party. In the execution of π, the view of parties are

$$\mathbf{view}_1^{\pi}(x,y,\kappa) = (x, r_1; m_1, m_2, \cdots, m_t), \tag{5.11}$$

$$\mathbf{view}_2^{\pi}(x,y,\kappa) = (y, r_2; m_1, m_2, \cdots, m_t), \tag{5.12}$$

where r_1, r_2 are the randomness of the parties, κ is the security parameter and m_i's are the intermediary messages received by each party. The output of π is $\mathbf{output}^{\pi}(x,y,\kappa) = (\mathbf{output}_1^{\pi}(x,y,\kappa), \mathbf{output}_2^{\pi}(x,y,\kappa))$, such that $\mathbf{output}_1^{\pi}(x,y,\kappa)$ and $\mathbf{output}_2^{\pi}(x,y,\kappa)$ are the local outputs of P_1 and P_2. We say that π securely computes $f(x,y)$ in the presence of semi-honest, non-adaptive, computationally bounded adversaries, if there exist probabilistic polynomial-time simulators $\mathscr{S}_1$ and $\mathscr{S}_2$ such that

$$\left\{\mathscr{S}_1(1^{\kappa}, x, f_1(x,y)), f(x,y)\right\} \equiv \left\{\mathbf{view}_1^{\pi}(x,y,\kappa), \mathbf{output}^{\pi}(x,y,\kappa)\right\}, \tag{5.13}$$

$$\left\{\mathscr{S}_2(1^{\kappa}, y, f_2(x,y)), f(x,y)\right\} \equiv \left\{\mathbf{view}_2^{\pi}(x,y,\kappa), \mathbf{output}^{\pi}(x,y,\kappa)\right\}. \tag{5.14}$$

Accordingly, Algorithm 13 is a protocol π between a server and a client which computes the functionality f that switches the computations from PHE to 2PC. The client does not provide an input for π (i.e. his input is an empty string $\perp$) apart from the auxiliary inputs encryption and decryption keys (pk, sk). The server's input is an ℓ-bit value x which is encrypted under the Paillier cryptosystem $[x]$. Given $[x]$, f computes $f(\perp, [x]) = (\langle x \rangle_c, \langle x \rangle_s)$ which are secret shares of x for the client and the server.

Theorem 5.4.1. The switching protocol π (Algorithm 13) securely computes the functionality $f(\perp, [x]) = (\langle x \rangle_c, \langle x \rangle_s)$ in the presence of semi-honest, non-adaptive, computationally bounded adversaries.

Proof. In the following, we prove Theorem 5.4.1 for a corrupted server and client separately, by showing that the view of adversary $\mathscr{A}$ in the real world is computationally indistinguishable from the simulated views of $\mathscr{S}_i$, where $i \in \{c, s\}$ is for the client and the server.

- **Server is corrupted by $\mathscr{A}$**: $\mathscr{S}_s$ is given the input and output of the server which are $[x], \langle x \rangle_s$, and the security parameter 1^κ. In simulation, we need to show that $\mathscr{S}_s$ can generate the view of incoming messages to the server, which is $\langle x + r \rangle_s$. $\mathscr{S}_s$ works as follows:

  1. $\mathscr{S}_s$ chooses a uniformly distributed random tape, r_1.
  2. $\mathscr{S}_s$ picks an $\ell + \kappa$-bit random value r' using the random tape r_1.
  3. $\mathscr{S}_s$ creates the secret shares $\langle r' \rangle_c$ and $\langle r' \rangle_s$.
  4. Using the output $\langle x \rangle_s$, $\mathscr{S}_s$ computes $\langle x + r' \rangle_s = \langle x \rangle_s + \langle r' \rangle_s$.

  The view of the server in the real world is

  $$\mathbf{view}_s^\pi(\perp, [x]) = ([x], r_s; \langle x + r \rangle_s), \tag{5.15}$$

  while the view generated by the simulator

  $$\mathscr{S}_s(1^\kappa, [x], \langle x \rangle_s) = \left([x], r_1; \langle x + r' \rangle_s\right). \tag{5.16}$$

  Since $\mathscr{S}_s$ does not have access to the decryption key sk, it cannot simulate $\mathrm{decr}([x + r])$. On the other hand, it can generate the intermediary message $\langle x + r' \rangle_s$, but if r' is uniformly sampled from r_1, then

  $$\left\{\mathscr{S}_s(1^\kappa, [x], \langle x \rangle_s), f(\perp, [x])\right\} \equiv \left\{\mathbf{view}_s^\pi(\perp, [x]), \mathbf{output}^\pi(\langle x \rangle_c, \langle x \rangle_s)\right\}, \tag{5.17}$$

  if for every nonuniform polynomial time distinguisher D there exists a negligible function $\mu(\kappa)$ such that

  $$\begin{aligned} &\Big|\Pr\left[D\left(([x], r_1; \langle x + r' \rangle_s) \wedge (\langle x \rangle_c, \langle x \rangle_s)\right) = 1\right] - \\ &\Pr\left[D(([x], r_s; \langle x + r \rangle_s) \wedge (\langle x \rangle_c, \langle x \rangle_s)) = 1\right]\Big| \leq \mu(\kappa). \end{aligned} \tag{5.18}$$

  Equation 5.18 holds due to the security of secure two-party computation and the uniformity of the random tape. The indistinguishability guarantees that a corrupted server has no advantage on differentiating $\langle x + r' \rangle_s$ from $\langle x + r \rangle_s$.

- **Client is corrupted by $\mathscr{A}$**: Different from the server, the client does not have an input for π. $\mathscr{S}_c$ is only provided the output $\langle x \rangle_c$ and the public and private keys pk, sk. To simulate the intermediary messages $[x + r]$ and $\langle r \rangle_c$, $\mathscr{S}_c$ works as follows:

  1. $\mathscr{S}_c$ chooses uniformly distributed random tapes r_1 and r_2.
  2. $\mathscr{S}_c$ picks an ℓ-bit random value x' and an $(\ell + \kappa)$-bit random value r' using the random tapes r_1, r_2.

3. $\mathscr{S}_c$ encrypts $x' + r'$ as $[x' + r']$ using the public key pk.
4. $\mathscr{S}_c$ creates secret shares for r' such that $r' \rightarrow \langle r' \rangle_c + \langle r' \rangle_s$.

The view of the client in the real world and the view generated by the simulator are

$$\mathbf{view}_c^{\pi}(\perp, [x]) = (\perp, r_c; [x + r], \langle r \rangle_c), \tag{5.19}$$

$$\mathscr{S}_c(1^{\kappa}, \perp, \langle x \rangle_c) = \left(\perp, r_1, r_2; [x' + r'], \langle r' \rangle_c\right), \tag{5.20}$$

respectively. Then,

$$\left\{\mathscr{S}_c(1^{\kappa}, \perp, \langle x \rangle_c), f(\perp, [x])\right\} \equiv \left\{\mathbf{view}_c^{\pi}(\perp, [x]), \mathbf{output}^{\pi}(\langle x \rangle_c, \langle x \rangle_s)\right\} \tag{5.21}$$

in the existence of a negligible function $\mu(\kappa)$ for every nonuniform polynomial time distinguisher D such that

$$\Big|\Pr\left[D\left(\left(\perp, r_1, r_2; [x' + r'], \langle r' \rangle_c\right) \wedge (\langle x \rangle_c, \langle x \rangle_s)\right) = 1\right] - \Pr\left[D\left((\perp, r_c; [x + r], \langle r \rangle_c) \wedge (\langle x \rangle_c, \langle x \rangle_s)\right) = 1\right]\Big| \leq \mu(\kappa). \tag{5.22}$$

Equation 5.22 is correct when a semantically secure encryption scheme and secret sharing scheme are used in securing messages $[x + r]$ and $\langle r \rangle_c$ which eliminate the advantage of distinguishing $[x + r]$ from $[x' + r']$ and $\langle r \rangle_c$ from $\langle r' \rangle_c$. Using the Paillier encryption scheme, which satisfies the semantic security under the decisional composite residuosity assumption, and arithmetic secret sharing, which guarantees information theoretic security, a corrupted client cannot break the indistinguishability. Furthermore, the adversary cannot reveal any information about x from the decryption of $[x + r]$, given that a sufficiently large, uniformly random value ($\ell + \kappa$ bits) is selected for masking x.

□

5.5. PERFORMANCE EVALUATION

We implemented SwaNN to evaluate its performance in different settings and to compare it with the state-of-the-art. We used the C++ programming language for the implementation and GMP 6.1.2 library for big integer operations. We used the ABY framework [19] for secure two-party computation operations. For the homomorphic operations, we used the Paillier implementation of ABY due to its efficiency. We selected 2048 bits modulus size in Paillier operations to meet the current security standards. For the ABY operations we selected 32-bit shares. The machine we used in the experiments runs Ubuntu 16.04 operating system with Intel Core i5-3470 CPU@3.20GHz.

5.5.1. OPTIMIZING COMPUTATIONS

In each layer of neural networks, the same operations are repeated for each index of the input independently. Thus, in our implementation we use several optimization techniques which help reduce the computation time and communication usage by enabling

simultaneous execution. To optimize 2PC computations, we use single instruction multiple data (SIMD) techniques [41] which are provided in the ABY framework. SIMD techniques cannot be fully utilized for the computations with the Paillier cryptosystem. Therefore, to improve the efficiency in homomorphic encryption, we adapt two techniques to the Paillier encryption which enables simultaneous computation.

The first technique we use is data packing. It packs multiple data items into a single ciphertext as described in [42]. Accordingly, we create slots of $t+\kappa$ bits for each data item where κ is the security parameter and t is the length of the data item. Given the plaintext modulus N, we can pack $\rho = \left\lfloor \frac{\log_2 N}{t+\kappa} \right\rfloor$ items in a single ciphertext as in Equation 5.23.

$$[\hat{x}] = \sum_{m=0}^{\rho-1} [x_{i,j}] \cdot (2^{t+\kappa})^m \tag{5.23}$$

Using data packing we can use the full plaintext domain in the Paillier cryptosystem and perform additions on the packed ciphertext simultaneously. Furthermore, in interactive protocols, using data packing helps reduce the bandwidth usage and the cost of decryption operations.

The second technique we use to improve efficiency of homomorphic encryption is using a multi-exponentiation algorithm to simultaneously perform the operations in the form of

$$\prod_{i=1}^{w} a_i^{b_i} = a_1^{b_1} \cdot a_2^{b_2} \dots a_w^{b_w}. \tag{5.24}$$

Lim-Lee's multi-exponentiation algorithm [43, 44] enables to perform Equation 5.24 simultaneously by modifying the binary exponentiation algorithm using several precomputation techniques. In our work, we can apply multi-exponentiation for the computation of dot product (Equation 5.9) over encrypted data thanks to the additive homomorphism of the Paillier cryptosystem. We summarize the optimizations used in each layer of neural networks as follows:

- `Conv`: Multi-exponentiation technique is used to reduce the cost of dot products.
- `Act`: Data packing is used before performing the activation function. If activation is performed with 2PC operations, then SIMD optimization is used.
- `Pool`: No optimization technique is needed.
- `FC`: Multi-exponentiation technique is used to reduce the cost of matrix multiplications.

5.5.2. EXPERIMENTS

We design two experiments with respect to the activation function used in the neural network. In the first experiment we used x^2 as the activation function and re-trained the neural network structure used in CryptoNets [8]. In the second experiment we used ReLU as the activation function and re-trained the neural network structure used in MiniONN [13]. The properties of the neural networks are detailed in Appendix 5.6.

EXPERIMENT 1

In the first experiment, we measured the performance of SwaNN with x^2 activation function in the client-server and the two-server scenario. For each scenario, we designed two different cryptographic setting. The first setting is an **only-PHE** setting which is totally based on the Paillier cryptosystem. We implemented the activation function x^2 as described in Algorithm 12. The second setting is a **hybrid setting** where the computation switches between PHE and 2PC. We implemented the secure switching protocols in Algorithm 13 and Algorithm 14 for this setting and implemented x^2 using the ABY framework. Table 5.3 demonstrates the performance of SwaNN for both scenarios in the only-PHE and the hybrid setting for each layer of the network. For the only-PHE setting we provide the timings with and without optimizations. For the hybrid setting, we provide only optimized timing values.

The results show that in the client-server scenario when no optimizations are used, the prediction of one image is computed approximately in 43 seconds. However, when we use optimization techniques, we can reduce the computation time to 27 seconds. In a hybrid setting, this cost is reduced to 10 seconds. Furthermore, in the two-server scenario with a slight increase in computation time, two images can be processed simultaneously. More particularly in an optimized hybrid setting the two servers can compute the prediction result for two images in 10 seconds simultaneously.

Table 5.2: Detailed computation time for the activation layer in the client-server and the two-server scenario for the hybrid setting (in ms).

Operation	Client	Server	Server-1	Server-2
Packing	–	3544	3551	3553
Decryption	73	–	142	146
Unpacking	0.1	–	0.1	
ABY	11	14	27	27
Encryption	2282	156	2358	2437
Total	6069		6078*	

In Table 5.2 we provide the details of the computation time for the activation layer in the hybrid setting. The packing, decryption and unpacking operations are performed during the switching from PHE to 2PC. The encryptions are computed by both parties when switching the operations from 2PC to PHE. In the client-server scenario, the client spends 2.36 seconds for the computations while the server spends approximately 3.7 seconds. In the two-server scenario, both servers spend approximately 6 seconds for the computation of the activation layer of two images.

Apart from computation time, we also analyzed the bandwidth usage of SwaNN for different settings. Table 5.4 shows the communication cost in both scenarios for the only-PHE setting and the hybrid setting. The packing technique used in the activation layers helps reduce the bandwidth usage by half. Besides due to the interactive nature of 2PC, the bandwidth usage in the hybrid setting is higher than the only-PHE setting for both scenarios.

Table 5.3: Computation time per layer in the client-server and the two-server scenario (in ms). The timings are provided for optimized and non-optimized PHE-only setting and optimized hybrid setting. The total timings marked with * show the simultaneous run time of SwaNN for two images.

	Non-optimized - PHE only				Optimized - PHE only				Optimized - Hybrid			
Layer	Client	Server	Server-1	Server-2	Client	Server	Server-1	Server-2	Client	Server	Server-1	Server-2
`Conv`	–	1873	1850	1857	–	372	377	369	–	371	370	372
`Act`	12629	15754	32680	32536	2484	19038	23494	23328	2366	3714	6078	6164
`Pool`	–	35	32	32	–	34	32	32	–	33	32	32
`Conv`	–	2852	2843	2831	–	550	566	547	–	548	548	550
`Pool`	–	35	35	35	–	35	37	35	–	35	35	35
`FC`	–	6395	6333	6319	–	2238	2227	2211	–	2216	2219	2234
`Act`	1496	1864	3846	3847	311	2207	2758	2758	273	501	798	786
`FC`	–	9	9	9	–	9	9	9	–	9	9	9
Total	42915		47466*		27248		29500*		10066		10182*	

Table 5.4: Bandwidth usage of SwaNN in different settings (in MB).

	Client-Server	Two-Server
PHE only (w/o opt.)	0.97	0.96
PHE only (w/ opt.)	0.51	0.51
Hybrid (w/ opt.)	1.63	1.73

As a final analysis, in Table 5.5 we compare SwaNN with the state-of-the-art works CryptoNets [8] and MiniONN [13] with respect to computation time and bandwidth usage. CryptoNets, which uses fully homomorphic encryption for computations, requires 297.5 seconds for one prediction. The protocol enables simultaneous computation by packing 4096 images into a single ciphertext. This is an advantage when the same client has very large number of prediction requests. MiniONN can compute the prediction result for the same network in 1.28 seconds. However, this computation requires 47.6 MB bandwidth usage. SwaNN can compute the same prediction result in 10 seconds. Although the computation time of SwaNN is higher than MiniONN, SwaNN achieves a 27-fold less bandwidth usage.

Table 5.5: Comparison with the state-of-the-art in Experiment 1.

	Computation time (s)	Bandwidth usage (MB)
CryptoNets [8]	297.5	372.2
MiniONN [13]	1.28	47.6
SwaNN	10.1	1.73

EXPERIMENT 2

As the second experiment, we measured the performance of SwaNN with ReLU activation function for the network described in Table 5.9 in Appendix 5.6. We used maximum operation for pooling layers. We provide the timings for the max pooling along with ReLU function since we implemented them together. We measure the timings in the client-server and the two-server scenario only with optimizations. Table 5.6 details the computation time for each layer. Due to larger number of input size in each layer of the network, the computation cost of SwaNN reaches to 61 seconds. The first activation layer is the dominant layer in the run time. As expected, the high computation cost is caused by the decryption operations which are performed during the switching phase from PHE to 2PC.

In Table 5.7, we compare the performance of SwaNN with MiniONN. Clearly, MiniONN outperforms SwaNN almost 7-fold in computation time. However, in terms of communication, SwaNN is more efficient with a bandwidth usage of 228 MB (compared to 657 MB in MiniONN).

Table 5.6: Computation time per layer in the client-server and the two-server scenario (in ms).

Layer	Client	Server	Server-1	Server-2
`Conv`	–	4118	4102	4098
`Act+Pool`	6855	46795	48777	48869
`Conv`	–	460	457	457
`Act+Pool`	766	4418	5602	5597
`FC`	–	1318	1329	1331
`Act`	277	506	815	815
`FC`	–	6	6	6
Total	57824		61173	

Table 5.7: Comparison with the state-of-the-art in Experiment 2.

	Computation time (s)	Bandwidth usage (MB)
MiniONN [13]	9.32	657.5
SwaNN	61.17	228.1

5.6. CONCLUSION

We have proposed a privacy preserving neural network prediction protocol that combines the additively homomorphic Paillier encryption scheme with secure two-party computation. Thanks to the use of the Paillier encryption algorithm for linear operations and also the x^2 activation function, the solution achieves better computational cost compared to existing HE-based solutions. Different computation optimisations based on the use of data packing and the multi-exponentiation algorithm have been implemented. Furthermore, the communication cost is also minimized since 2PC is only used for non-linear operations (max pooling and/or RELU). SwaNN can be executed in the two-server setting, in case the client lacks resources. Experimental results show that SwaNN actually achieves the best of both worlds, namely, better computational overhead compared to HE-based solutions and, better communication overhead compared to 2PC-based solutions.

NEURAL NETWORK STRUCTURES

In our experiments, we use two neural network structures which are previously trained by CryptoNets [8] and MiniONN [13] to perform image classification on MNIST data. Table 5.8 summarizes the structure of the neural network proposed in CryptoNets. The accuracy of the networks is 98.95%. The network has 9 layers. Since the last layer of the network, the sigmoid activation, is applied only in the training phase, we did not include it in our experiments. The activation function of the network is x^2. As pooling operation, scaled mean pooling is used. Secondly, we used the neural network structure proposed in MiniONN (Figure 12) [13]. The accuracy of the network is 99.31%. Table 5.9

demonstrates the layers of the network. The activation function of the network is ReLU. Max pooling is used in the pooling layer.

Table 5.8: CryptoNets Neural Network structure [8].

Layer	Input size	Output size	Filter	Stride
`Conv`	28×28	$5 \times 13 \times 13$	5×5	(2,2)
`Act`	$5 \times 13 \times 13$	$5 \times 13 \times 13$		
`Pool`	$5 \times 13 \times 13$	$5 \times 13 \times 13$	3×3	1
`Conv`	$5 \times 13 \times 13$	$50 \times 5 \times 5$	5×5	(2,2)
`Pool`	$50 \times 5 \times 5$	$50 \times 5 \times 5$	3×3	1
`FC`	$50 \times 5 \times 5$	100×1		
`Act`	100×1	100×1		
`FC`	100×1	10×1		

Table 5.9: MiniONN Neural Network structure [13].

Layer	Input size	Output size	Filter	Stride
`Conv`	28×28	$16 \times 24 \times 24$	5×5	(1,1)
`Act`	$16 \times 24 \times 24$	$16 \times 24 \times 24$		
`Pool`	$16 \times 24 \times 24$	$16 \times 12 \times 12$	2×2	2
`Conv`	$16 \times 12 \times 12$	$16 \times 8 \times 8$	5×5	(1,1)
`Act`	$16 \times 8 \times 8$	$16 \times 8 \times 8$		
`Pool`	$16 \times 8 \times 8$	$16 \times 4 \times 4$	2×2	2
`FC`	$16 \times 4 \times 4$	100×1		
`Act`	100×1	100×1		
`FC`	100×1	10×1		

REFERENCES

[1] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff, *Privacy-preserving classification on deep neural network,* IACR Cryptology ePrint Archive **2017**, 35 (2017).

[2] K. Fukushima, S. Miyake, and T. Ito, *Neocognitron: A neural network model for a mechanism of visual pattern recognition,* IEEE Trans. Systems, Man, and Cybernetics **13**, 826 (1983).

[3] D. C. Ciresan, U. Meier, and J. Schmidhuber, *Multi-column deep neural networks for image classification,* CoRR **abs/1202.2745**, 20 (2012).

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Imagenet classification with deep convolutional neural networks,* in *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.* (2012) pp. 1106–1114.

[5] F. Schroff, D. Kalenichenko, and J. Philbin, *Facenet: A unified embedding for face recognition and clustering,* in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015* (IEEE Computer Society, 2015) pp. 815–823.

[6] N. Jones, *Computer science: The learning machines,* Nature: Computer Science **505**, 146 (2014).

[7] M. Ribeiro, K. Grolinger, and M. A. M. Capretz, *Mlaas: Machine learning as a service,* in *14th IEEE International Conference on Machine Learning and Applications, ICMLA 2015, Miami, FL, USA, December 9-11, 2015* (IEEE, 2015) pp. 896–902.

[8] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. E. Lauter, M. Naehrig, and J. Wernsing, *Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy,* in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016* (JMLR.org, 2016) pp. 201–210.

[9] M. Barni, C. Orlandi, and A. Piva, *A privacy-preserving protocol for neural-network-based computation,* in *Proceedings of the 8th workshop on Multimedia & Security, MM&Sec 2006, Geneva, Switzerland, September 26-27, 2006* (ACM, 2006) pp. 146–151.

[10] C. Orlandi, A. Piva, and M. Barni, *Oblivious neural network computing via homomorphic encryption,* EURASIP J. Information Security **2007**, 1 (2007).

[11] P. Mohassel and Y. Zhang, *Secureml: A system for scalable privacy-preserving machine learning,* in *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017* (IEEE Computer Society, 2017) pp. 19–38.

[12] P. Mohassel and P. Rindal, *Aby^3: A mixed protocol framework for machine learning,* in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018* (ACM, 2018) pp. 35–52.

[13] J. Liu, M. Juuti, Y. Lu, and N. Asokan, *Oblivious neural network predictions via minionn transformations,* in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017* (ACM, 2017) pp. 619–631.

[14] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, *GAZELLE: A low latency framework for secure neural network inference,* in *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018.* (USENIX Association, 2018) pp. 1651–1669.

[15] P. Paillier, *Public-key cryptosystems based on composite degree residuosity classes,* in *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding* (Springer, 1999) pp. 223–238.

[16] I. Damgård and M. Jurik, *A generalisation, a simplification and some applications of paillier's probabilistic public-key system,* in *Public Key Cryptography, 4th International Workshop on Practice and Theory in Public Key Cryptography, PKC 2001, Cheju Island, Korea, February 13-15, 2001, Proceedings* (Springer, 2001) pp. 119–136.

[17] D. Beaver, *Efficient multiparty protocols using circuit randomization,* in *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings* (Springer, 1991) pp. 420–432.

[18] O. Goldreich, S. Micali, and A. Wigderson, *How to play any mental game or A completeness theorem for protocols with honest majority,* in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA* (ACM, 1987) pp. 218–229.

[19] D. Demmler, T. Schneider, and M. Zohner, *ABY - A framework for efficient mixed-protocol secure two-party computation,* in *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015* (The Internet Society, 2015) pp. 1–15.

[20] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, *Deepsecure: scalable provably-secure deep learning,* in *Proceedings of the 55th Annual Design Automation Conference, DAC 2018, San Francisco, CA, USA, June 24-29, 2018* (ACM, 2018) pp. 2:1–2:6.

[21] M. S. Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar, *Chameleon: A hybrid secure computation framework for machine learning applications,* in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security, AsiaCCS 2018, Incheon, Republic of Korea, June 04-08, 2018* (ACM, 2018) pp. 707–721.

[22] M. Dahl, J. Mancuso, Y. Dupis, B. Decoste, M. Giraud, I. Livingstone, J. Patriquin, and G. Uhma, *Private machine learning in tensorflow using secure computation,* CoRR **abs/1810.08130** (2018).

[23] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, *TensorFlow: Large-scale machine learning on heterogeneous systems,* (2015), software available from tensorflow.org.

[24] S. Wagh, D. Gupta, and N. Chandran, *Securenn: Efficient and private neural network training,* in *Privacy Enhancing Technologies Symposium* ((PETS 2019), 2019).

[25] SEAL, *Simple Encrypted Arithmetic Library (release 3.1.0),* (2018), microsoft Research, Redmond, WA.

[26] E. Hesamifard, H. Takabi, and M. Ghasemi, *Cryptodl: Deep neural networks over encrypted data,* CoRR **abs/1711.05189** (2017), arXiv:1711.05189 .

[27] A. Ibarrondo and M. Önen, *Fhe-compatible batch normalization for privacy preserving deep learning,* in *Data Privacy Management, Cryptocurrencies and Blockchain Technology - ESORICS 2018 International Workshops, DPM 2018 and CBT 2018, Barcelona, Spain, September 6-7, 2018, Proceedings* (Springer, 2018) pp. 389–404.

[28] F. Bourse, M. Minelli, M. Minihold, and P. Paillier, *Fast homomorphic evaluation of deep discretized neural networks,* in *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part III* (Springer, 2018) pp. 483–512.

[29] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachene, *Tfhe: Fast fully homomorphic encryption over the torus,* Cryptology ePrint Archive, Report 2018/421 (2018), https://eprint.iacr.org/2018/421.

[30] A. Sanyal, M. J. Kusner, A. Gascón, and V. Kanade, *TAPAS: tricks to accelerate (encrypted) prediction as a service,* CoRR **abs/1806.03461** (2018).

[31] E. Hesamifard, H. Takabi, M. Ghasemi, and R. N. Wright, *Privacy-preserving machine learning as a service,* PoPETs **2018**, 123 (2018).

[32] X. Jiang, M. Kim, K. E. Lauter, and Y. Song, *Secure outsourced matrix computation and application to neural networks,* in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018* (ACM, 2018) pp. 1209–1222.

[33] E. Chou, J. Beal, D. Levy, S. Yeung, A. Haque, and L. Fei-Fei, *Faster cryptonets: Leveraging sparsity for real-world encrypted inference,* CoRR **abs/1811.09953** (2018).

[34] O. Ohrimenko, F. Schuster, C. Fournet, A. Mehta, S. Nowozin, K. Vaswani, and M. Costa, *Oblivious multi-party machine learning on trusted processors,* in *25th USENIX Security Symposium (USENIX Security 16)* (2016) pp. 619–636.

[35] T. Hunt, C. Song, R. Shokri, V. Shmatikov, and E. Witchel, *Chiron: Privacy-preserving machine learning as a service,* CoRR **abs/1803.05961** (2018).

[36] F. Tramèr and D. Boneh, *Slalom: Fast, verifiable and private execution of neural networks in trusted hardware,* in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019* (2019).

[37] T. Toft, *Sub-linear, secure comparison with two non-colluding parties,* in *Public Key Cryptography - PKC 2011 - 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings* (Springer, 2011) pp. 174–191.

[38] A. C. Yao, *Protocols for secure computations (extended abstract),* in *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982* (IEEE Computer Society, 1982) pp. 160–164.

[39] W. Henecka, S. Kögl, A. Sadeghi, T. Schneider, and I. Wehrenberg, *TASTY: tool for automating secure two-party computations,* in *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010* (ACM, 2010) pp. 451–462.

[40] Y. Lindell, *How to simulate it - A tutorial on the simulation proof technique,* in *Tutorials on the Foundations of Cryptography.* (Springer International Publishing, 2017) pp. 277–346.

[41] N. P. Smart and F. Vercauteren, *Fully homomorphic SIMD operations,* Des. Codes Cryptography **71**, 57 (2014).

[42] T. Bianchi, A. Piva, and M. Barni, *Composite signal representation for fast and storage-efficient processing of encrypted signals,* IEEE Trans. Information Forensics and Security **5**, 180 (2010).

[43] C. H. Lim and P. J. Lee, *More flexible exponentiation with precomputation,* in *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings* (Springer, 1994) pp. 95–107.

[44] C. H. Lim, *Efficient multi-exponentiation and application to batch verification of digital signatures,* Unpublished manuscript, August **n.a.**, 1 (2000).

6

PRIVACY-PRESERVING ONLINE BEHAVIOURAL ADVERTISING

Online advertising is a multi-billion dollar industry, forming the primary source of income for many publishers offering free web content. Serving advertisements tailored to users' interests greatly improves the effectiveness of advertisements, and is believed to be beneficial to publishers and users alike. The privacy of users, however, is threatened by the widespread collection of data that is required for behavioural advertising. In this chapter, we present two privacy-preserving protocols for online behavioural advertising that combine machine learning methods with secure computation techniques. The first protocol uses a threshold variant of an additively homomorphic cryptosystem to distribute trust between parties while allowing computations on encrypted data, such that advertisements can be served based on detailed user profiles. The second protocol distributes trust between advertising companies using an additively homomorphic threshold secret sharing scheme, allowing collaborative computations on user profiles while preventing a coalition of colluding parties smaller than a predefined threshold from obtaining any sensitive information. Both protocols achieve performance multi-linear in the size of user profiles and the number of advertising campaigns, and show promising initial results in terms of privacy and performance. To the best of our knowledge, our two protocols are the first protocols that preserve user privacy in behavioural advertising while allowing the use of detailed user profiles and machine learning methods.

This chapter has been published as

- Section 6.1
  "*AHEad: Privacy-preserving Online Behavioural Advertising using Homomorphic Encryption*" by L. Helsloot, G. Tillem, and Z. Erkin in *IEEE Workshop on Information Forensics and Security, WIFS 2017* (pp. 1-6) (2017)
- Section 6.2
  "*BAdASS: Preserving Privacy in Behavioural Advertising with Applied Secret Sharing*" by L. Helsloot, G. Tillem, and Z. Erkin in *JoWUA* (pp.23-41) (2019).

6.1. AHEAD: PRIVACY-PRESERVING ONLINE BEHAVIOURAL ADVERTISING USING HOMOMORPHIC ENCRYPTION

Online advertising is a pervasive phenomenon on the Internet, backed by a multi-billion dollar industry with a worldwide spend of $178 billion in 2016 [1]. Advertisements allow publishers to offer web services free of charge, forming a primary financial pillar supporting free web content [2]. An increasing number of people object to being shown advertisements on web pages they visit, however, resulting in a rapid adoption of technological measures to block advertisements. In early 2017, it was estimated that ad blocking tools were installed on 615 million devices, amounting to 11% of the Internet population, and the adoption of such tools is predicted to increase in the future [3]. The use of ad blockers has lead to a significant loss of revenue from advertising space offered by publishers. The worldwide cost of ad blocking, in terms of missed revenue, was estimated to be $41.4 billion, or 23% of the total ad spend, in 2016 [4]. These developments threaten the business models of many free web services, necessitating measures to alleviate objections against advertising in order to attain a sustainable advertisement-supported Internet economy.

One of the objections people have to online advertisements is that widespread data collection by advertising companies infringes on user privacy [5]. 32% of respondents to a recent survey among ad blocker users indicated that privacy concerns were a reason for their use of an ad blocker [6]. In a similar survey on privacy and advertising, 94% of respondents indicated that online privacy was an important issue, and according to 70% of respondents, online advertising networks and online advertisers should be responsible for users' online privacy [7]. The widespread collection of user data that sparks privacy concerns is a key element of behavioural targeting, in which the advertisements that are shown to a user are selected based on the user's browsing behaviour. Although such tailored advertisements are recognized by users as being useful for both publishers and users, acceptance of behavioural advertising is hindered by a mistrust of advertising companies and a lack of control over the collection of information [5].

ONLINE BEHAVIOURAL ADVERTISING

Online Behavioural Advertising (OBA) is the practice of serving advertisements based on individuals' interests. These interests are inferred from users' browsing behaviour, using data such as visited web pages, search queries, and online purchases. OBA allows for advertisements to be targeted at individual users, greatly improving the Click-Through Rate (CTR) and thus the expected effectiveness of advertisements [8]. Personalization of advertisements is typically performed using campaign-specific supervised machine learning models that predicts users' responses to advertisements. Users benefit from OBA by being served less irrelevant advertisements, whereas advertisers benefit from accurate targeting as it allows them to reach the desired audience. Finally, publishers experience an increased value of the advertising space they offer.

The current trend in OBA is the Real-Time Bidding (RTB) mechanism of buying and selling ads [9]. In the RTB model, ad exchanges (AdX) provide marketplaces where advertising space is auctioned in real time for individual impressions. When a user visits a web page containing advertising space, the publisher offers the ad impression for bids at one or more ad exchanges, after which advertisers place their bids within a fraction

of a second. Other platforms emerged along with ad exchanges to manage the complexity of RTB. Demand-Side Platforms (DSPs) bid on impressions from multiple inventories on behalf of advertisers, who may not possess the expertise required to partake in programmatic real-time auctions. Supply-Side Platforms (SSPs) assist publishers in reaching a large number of advertisers by offering advertising space to multiple inventories.

Existing literature proposes a number of methods to address privacy concerns in OBA. These methods include blocking advertisements altogether [10], obfuscating browsing behaviour [11], and anonymization [12], as well as exposing only generalized user profiles to advertising companies [13]. However, limiting the data available to advertising companies is expected to decrease the accuracy of targeted advertisements [14], and thus the value of advertisements to users, advertisers, and publishers. Finally, some work proposes cryptographic approaches to aggregate ad click statistics [13] or perform advertisement selection using secure hardware [15]. These approaches, however, assume the existence of centralized advertising networks performing simple keyword-based advertisement selection, and are thus unsuitable for use within the highly distributed RTB model.

OUR CONTRIBUTIONS

In this paper, we present AHEad, a novel protocol that preserves user privacy in OBA, is compatible with the RTB mechanism of buying ads, and supports highly detailed user profiles. To the best of our knowledge, this is the first protocol making use of machine learning on encrypted data for preserving user privacy in OBA tasks. AHEad is based upon machine learning techniques commonly encountered in existing OBA systems, and allows multiple data processors, specifically DSPs, to operate on the same encrypted user data. Our protocol distributes trust between parties using threshold homomorphic encryption, such that no single party can decrypt sensitive information. We achieve performance linear in the size of user profiles, and see room for further performance improvements.

6.1.1. PRELIMINARIES

LOGISTIC REGRESSION

Although a variety of machine learning algorithms has been proposed for user response estimation tasks, logistic regression has recently been used in production systems by many advertising companies, such as Google [16], Facebook [17], and Criteo [18], and is considered a state-of-the-art model [19]. A logistic regression model is used to estimate the probability of a binary outcome y (click or no click) based on a d-dimensional predictor vector $\boldsymbol{x}$ (the user profile) [9]. Given a feature vector $\boldsymbol{x}$ and model parameters $\boldsymbol{w}$, the model predicts the click probability $\hat{y}$ using the sigmoid function:

$$\hat{y} = \sigma(\boldsymbol{w}^\top \boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{w}^\top \boldsymbol{x}}}. \tag{6.1}$$

After observing the actual binary click label y, we use the gradient of the logistic loss to update model parameters $\boldsymbol{w}$ using an online Stochastic Gradient Descent (SGD) method as in [16]. Given the gradient of the loss $\boldsymbol{g} = (\hat{y} - y)\boldsymbol{x}$, we perform the model update $\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \boldsymbol{g}$, where η is the learning rate. Note that the learning rate can be a constant, but may also be set to decay per iteration and per coordinate of the gradient.

FEATURE HASHING

Many of the features considered in OBA are categorical. The high cardinality of some of these features results in a very large feature space, particularly if feature conjunctions are used. To encode categories from a user profile into a vector of fixed dimensionality for use in logistic regression, we use the hashing trick [20], which showed promising results in e.g. [18]. The hashing trick maps values from our high-dimensional user profile into a lower-dimensional vector $\boldsymbol{x}$ by setting x_i to a count of the values whose hash is i. The resulting d-dimensional vector $\boldsymbol{x}$ (in [18], $d = 2^{24}$ is used) is the input feature vector to the logistic regression model.

THRESHOLD HOMOMORPHIC ENCRYPTION

Our protocol relies on an additively homomorphic cryptosystem, such as Paillier [21]. The homomorphic properties of Paillier allow the computation of

$$\mathcal{E}_{pk}(m_1 + m_2) = \mathcal{E}_{pk}(m_1) \cdot \mathcal{E}_{pk}(m_2)$$

by any party with access to public key pk, where $\mathcal{E}_{pk}(x)$ is the encryption of a message x under public key pk. Likewise, one can compute $\mathcal{E}_{pk}(c \cdot m_1) = \mathcal{E}_{pk}(m_1)^c$ for a public constant c.

In our protocol, we use a two-party threshold version of Paillier as described in [22], such that trust is distributed between parties. We denote an encryption using threshold Paillier as $[\cdot]$, and have omitted share decryption steps from our protocol descriptions. We use $[\cdot]_u$ and $[\cdot]_{DSP}$ to represent encryptions using any asymmetric encryption scheme under the public key of the user or DSP, respectively.

6.1.2. PROTOCOL DESIGN

For our privacy-preserving online advertising protocol, we consider a setting that is compatible with the existing RTB landscape. We assume that DSPs are the only parties that perform operations on user data, i.e. that personalization of advertisements is performed solely by DSPs. The majority of our protocol is therefore performed at the DSPs, and ad exchanges are responsible only for collecting bids and selecting the highest bid. SSPs are assumed to only offer advertising space to multiple ad exchanges, and are not considered in our protocol for simplification purposes.

Since some existing companies act as both ad exchange and DSP, we assume that ad exchanges and DSPs collude. To preserve privacy in the presence of colluding parties, we introduce an additional entity into our setting called Privacy Service Provider (PSP). The PSP is assumed not to collude with any party, but is not trusted with user data. Therefore, we use a threshold version of Paillier, where one share of the private key is held by the PSP, and each of the DSPs and ad exchanges holds a copy of the other share. An advertising company must thus collaborate with the PSP to decrypt and vice versa. We consider the following parties in our protocol:

- Users are the actors to whom advertisements are shown when they visit web pages. A user visits web pages using a web browser, which maintains a profile describing the user's behaviour. Whenever a web page containing advertising space is visited,

the web browser requests an advertisement from an ad exchange. After displaying the ad, the web browser sends click feedback to the ad exchange, indicating whether the ad was clicked or not.

- Ad exchanges offer a marketplace where advertising space is auctioned in real time. Ad exchanges send bid requests to DSPs, and pick the highest bid from the responses.
- DSPs bid on advertising space on behalf of one or more advertisers. For every received bid request, a DSP generates a bid for each of its campaigns by running a response prediction model on the received input, and submits these bids and the associated advertisement to the ad exchange.
- PSP is a service provider that assists in performing computations in a privacy-preserving manner.

Our protocol is based on a semi-honest security model, where ad exchanges and DSPs execute the protocol together with the PSP. The ad exchanges, DSPs and PSP should not learn any information about the contents of the user profile, nor which ads were viewed or clicked, from the protocol execution. Moreover, ad exchanges and the PSP should not learn any information about the parameters of the models run by DSPs, nor the bid values submitted by DSPs.

Table 6.1: Explanation of Symbols

Symbol	Explanation
$\boldsymbol{x}$	Input feature vector obtained by feature hashing.
K_γ	Set of campaigns run by a DSP γ.
$\boldsymbol{w}_k$	Model parameters for a campaign k, where $w_{k,i}$ is the i^{th} coordinate of $\boldsymbol{w}_k$.
$\eta_{k,i}$	Learning rate parameter for campaign k and coordinate i.
$\pi(\cdot)$	Random permutation function.
$\pi^{-1}(\cdot)$	Inverse permutation of $\pi(\cdot)$, such that $\pi^{-1}(\pi(\boldsymbol{x})) = \boldsymbol{x}$.
$B_k(\cdot)$	Bidding function for campaign k.
b_k	Bid value for campaign k.
a_k	Advertisement associated with campaign k.
k	Unique campaign identifier.

INITIAL SETUP

Prior to protocol execution, a key pair for the two-party Paillier scheme is generated, either by a Trusted Third Party (TTP) or using a distributed protocol as outlined in e.g. [22]. The private key is secret shared, such that the PSP holds one part of the key, and each DSP

and ad exchange holds a copy of the other part of the key. Moreover, each DSP generates a key pair for use in the profile update protocol. Finally, advertisers set up their campaigns such that DSPs can bid on their behalf.

USER PROFILING PHASE

Browsing behaviour is recorded within the user's web browser to form a local user profile, such that privacy can be preserved during the user profiling phase. Local profiling can be performed using existing techniques, such as RePriv [23]. The resulting profile information is captured into a d-dimensional feature vector $\boldsymbol{x}$ using feature hashing.

Since sending the full d-dimensional feature vector during each advertisement request would incur prohibitively high communication costs, feature vectors are cached at DSPs. Caching significantly reduces the amount of communication required during the time-sensitive advertisement selection phase, and allows feature vectors to be updated in the background to minimize delays experienced by the user. To further decrease communication costs, profile updates are periodically sent by the user. Depending on the expected size of user profiles, profile updates can be performed in either an incremental fashion or by completely replacing the user profile. Incremental updates may be less costly to encrypt and transmit for users, but require computationally expensive operations at DSPs. AHEad takes the latter approach of replacing the user profile. However, it can easily be modified to process incremental updates by using the additively homomorphic properties of the encryption scheme.

The steps performed during a periodic profile update are outlined in Algorithm 16. An explanation of the symbols used in the protocol descriptions is given in Table 6.1. The user with unique identifier u generates a d-dimensional feature vector $\boldsymbol{x}$ from their local profile information.

Due to feature hashing, $\boldsymbol{x}$ is expected to be a very sparse high-dimensional vector. To reduce the communication costs for the user, the profile update is encoded as a set of pairs $P = \{(i + r \pmod d, [x_i])\}$ for non-zero x_i, where $r \in_R \mathbb{Z}_d$. This compressed representation is sent to the PSP, along with encryption $[(u, r)]_{DSP}$. The PSP then expands P into an element-wise encryption of the original vector $\boldsymbol{x}$, with its elements rotated r times due to the randomization of indices performed by the user, setting any element not present in P to $[0]$. This expansion is sent to the relevant DSPs, along with $[(u, r)]_{DSP}$. The DSPs decrypt the user's identifier to select the relevant user profile, and the random number to rotate the received expansion back to its original indices. Finally, the DSPs replaces the previous profile information of the user with the encrypted feature vector.

BIDDING PHASE

During the bidding phase, every DSP calculates a bidding price for each of their campaigns, based on a cached user profile. The bidding phase is initiated by the user contacting an ad exchange with a request for an advertisement. The ad exchange sends a bid request to every DSP, each of which executes the bidding protocol. For each advertising campaign k, the user response $\hat{y}$ is assumed to be estimated using a logistic regression model. The bidding price is derived from $\hat{y}$ using a linear bidding function $B(\hat{y}_k) = c_1\hat{y}_k + c_2$ for campaign-specific constants c_1 and c_2. Although a linear bidding function may not be capable of fully capturing complex bidding strategies used in prac-

Algorithm 16 Profile update protocol, initiated by every user

Input: $\boldsymbol{x}, u$

Pick $r \in_R \mathbb{Z}_d$
$P \leftarrow \{(i + r \pmod d), [x_i]]) \mid x_i \neq 0\}$
invoke EXPAND$(P, [(u,r)]_{DSP})$ at PSP

procedure EXPAND$(P, [(u,r)]_{DSP})$
  for $j \leftarrow 1, d$ **do**
    $[\tilde{x}_j] \leftarrow \begin{cases} [v] & \text{if } (j, [v]) \in P \\ [0] & \text{otherwise} \end{cases}$
  invoke UPDATE$([\tilde{\boldsymbol{x}}], [(u,r)]_{DSP})$ at DSP

procedure UPDATE$([\tilde{\boldsymbol{x}}], [(u,r)]_{DSP})$
  Decrypt $[(u,r)]_{DSP}$
  Select profile $[\boldsymbol{x}]$ for user u
  for $i \leftarrow 1, d$ **do**
    $[x_i] \leftarrow [\tilde{x}_{(i-r \pmod d)}]$

tice, it is used for illustrative purposes, and could be replaced by another bidding function.

The bidding protocol, as executed by the DSPs, is outlined in Algorithm 17. The sigmoid function that is used to make predictions in logistic regression models is non-trivial to compute under additively homomorphic encryption. Existing literature uses two different approaches to compute the result of the sigmoid function: approximation [24, 25], or computation in the clear [26, 27]. We argue that in our setting, revealing the input to the sigmoid function $\boldsymbol{w}^\top \boldsymbol{x}$ to the PSP is acceptable, since it does not leak information about the user's profile as $\boldsymbol{w}$ is not known to the PSP. (Note that, if $\boldsymbol{w}$ is known to the PSP, it could be possible to extract information about $\boldsymbol{x}$. Similarly, if $\boldsymbol{x}$ is known to the PSP, it could be possible to extract information about $\boldsymbol{w}$.) The result of the sigmoid function may leak information about the degree to which the user is interested in a particular topic. However, no information about the identity of the user is passed to the PSP during the bidding phase, and thus the PSP can not infer any more information than that there is a user who is interested in a particular topic. Finally, the result of the sigmoid function does not reveal information about bid values to the PSP, as the bidding functions are unknown to the PSP.

AUCTION PHASE

During the auction phase, the PSP and the ad exchange engage in a secure comparison protocol, such as described in e.g. [28], to select the highest bid and associated advertisement without either party learning which bid won the auction. At the start of the auction phase the PSP holds encryptions of all bids submitted by the DSPs through the bidding protocol. These bids consist of encryptions of the bid value, the advertisement, the predicted response, and the campaign identifier. The bids are randomly permuted by the

Algorithm 17 Bidding protocol, initiated by every DSP

Input: $\{(\boldsymbol{w}_k, B_k, a_k) \mid k \in K_\gamma\}, [\boldsymbol{x}]$

1: **for all** $k \in K_\gamma$ **do**
2: $\quad [s_k] \leftarrow \prod_{i=1}^{d} [x_i]^{w_{k,i}}$
3: Pick random permutation function $\pi_s(\cdot)$
4: $[\boldsymbol{s}'] \leftarrow \pi_s([\boldsymbol{s}])$
5: **invoke** $[\hat{\boldsymbol{y}}'] \leftarrow$ CALCULATE-SIGMA$([\boldsymbol{s}'])$ at PSP
6: $[\hat{\boldsymbol{y}}] \leftarrow \pi_s^{-1}([\hat{\boldsymbol{y}}'])$
7: **for all** $k \in K_\gamma$ **do**
8: $\quad$ Re-randomize $[\hat{y}_k]$
9: $\quad [b_k] \leftarrow B_k([\hat{y}_k])$
10: $\quad$ **send** $[a_k]_u, [b_k], [\hat{y}_k], [k]$ to PSP

11: **procedure** CALCULATE-SIGMA$([\boldsymbol{s}'])$
12: $\quad$ Decrypt $[\boldsymbol{s}']$
13: $\quad$ **for all** $s'_i \in \boldsymbol{s}'$ **do**
14: $\quad\quad \hat{y}'_i \leftarrow \sigma(s'_i)$
15: $\quad$ **return** $[\hat{\boldsymbol{y}}']$

PSP and sent to the ad exchange. After execution of the secure comparison protocol, the ad exchange holds an encryption of the index of the highest bid, which is decrypted by the ad exchange. This allows the ad exchange to forward the encrypted bid information to the user, who decrypts and displays the advertisement.

MODEL UPDATE PHASE

After an advertisement is shown to a user, the response prediction model associated with the shown advertisement can be updated to learn from the observed user action y, which is either click (1) or no click (0). In the current non-privacy-preserving setting, only clicks are reported, and non-clicks are inferred from the absence of clicks. Since we want to hide whether a user clicked on an advertisement, however, we always report something. Note that we cannot reveal $\hat{y}-y$, since its value leaks information about y, and we cannot reveal $\hat{y}$ to the PSP as the PSP could link that to values observed during the bidding phase. Therefore, we must rely on users to calculate the gradients used for model updates.

The model update protocol is outlined in Algorithm 18. The user calculates the gradient of the loss function $\boldsymbol{g} = (\hat{y} - y)\boldsymbol{x}$ in the encrypted domain. DSPs have their model parameters $\boldsymbol{w}$ in the clear, so an update of $\boldsymbol{w}$ based on a single user's response reveals information about the user's profile $\boldsymbol{x}$ to the DSP. Therefore, the user sends $\boldsymbol{g}$ to the PSP, where gradients of multiple users are aggregated before revealing the resulting vector to the DSP. The DSP can then update the response prediction model based on the aggregated gradients of a small batch of users. Moreover, the PSP aggregates bid values received by users on a per-campaign basis, such that advertisers can be billed without revealing individual bid values.

Algorithm 18 Model update protocol, initiated by a user after viewing an advertisement

Input: $\boldsymbol{x}, y, [\hat{y}], [b], [k]$

1: $[\delta] \leftarrow [\hat{y}] \cdot [-y]$
2: **for** $i \leftarrow 1, d$ **do**
3: $\quad [g_i] \leftarrow \begin{cases} [\delta]^{x_i} & \text{if } x_i \neq 0 \\ [0] & \text{otherwise} \end{cases}$
4: Re-randomize $[b], [k]$
5: **invoke** AGGREGATE($[\boldsymbol{g}], [k], [b]$) at PSP

6: **procedure** AGGREGATE($[\boldsymbol{g}], [k], [b]$)
7: $\quad$ Decrypt $[k]$
8: $\quad$ **for** $i \leftarrow 1, d$ **do**
9: $\quad\quad [\hat{g}_{k,i}] \leftarrow [\hat{g}_{k,i}] \cdot [g_i]$
10: $\quad [\hat{b}_k] \leftarrow [\hat{b}_k] \cdot [b]$
11: $\quad$ **if** sufficient values are aggregated for campaign k **then**
12: $\quad\quad$ **invoke** UPDATE($[\hat{\boldsymbol{g}}_k], k$) at DSP

13: **procedure** UPDATE($[\boldsymbol{g}], k$)
14: $\quad$ Decrypt $[\boldsymbol{g}]$
15: $\quad$ **for** $i \leftarrow 1, d$ **do**
16: $\quad\quad w_{k,i} \leftarrow w_{k,i} - \eta_k g_i$

6.1.3. COMPUTATIONAL ANALYSIS

To evaluate the computational complexity of the proposed protocol, we provide both a theoretical analysis in terms of the number of cryptographic operations performed by parties participating in the protocol, and a set of measurements obtained from a proof-of-concept implementation.

Table 6.2: Symbols Used in Computational Analysis

Symbol	Description
ν	Number of non-zero elements in the user's profile.
d	Dimensionality of user profiles.
κ	Number of campaigns of a DSP.
K	Total number of campaigns.
ζ	Number of model updates aggregated per campaign.

COMPUTATIONAL COMPLEXITY

The variables used in the computational analysis are described in Table 6.2. Note that ν, κ, K and ζ are expected to be several orders of magnitude smaller than d. Table 6.3

lists the amortized number of operations performed by each party for each subprotocol. While the proposed protocol requires a large number of cryptographic operations, all subprotocols have a complexity at most linear in the number of campaigns and size of user profiles. The bidding protocol requires the largest number of operations due to the computation of $[\boldsymbol{w}^\top \boldsymbol{x}]$. However, this computation is trivially parallelizable and can thus be greatly sped up. Moreover, the required number of operations can be reduced by employing a sparsity-inducing model such as the Follow The (Proximally) Regularized Leader (FTRL) model described in [16]. Further speed improvements can be achieved by packing multiple values into a single ciphertext in the model update phase, reducing the number of encryptions performed by the user, the number of multiplications performed by the DSP and the number of share decryptions performed by the DSP and PSP. Moreover, packing reduces the amount of communication required between the parties.

Table 6.3: Number of Operations Performed per Subprotocol

Protocol	**Operation**	**User**	**AdX**	**DSP**	**PSP**
Profile update	Encryption	ν			$d-\nu$
	Decryption			1	
Bidding	Encryption			2κ	K
	Share decryption			κ	K
	Multiplication			$\kappa(d-1)$	
	Exponentiation			κd	
	Randomization			κ	
	Bidding function			κ	
Auction	Comparison		$K-1$		$K-1$
	Decryption		1		
Model update	Exponentiation	ν			
	Encryption	$d-\nu$			
	Multiplication	1			$1+d$
	Share decryption			$1+\frac{d}{\zeta}$	$1+\frac{d}{\zeta}$

IMPLEMENTATION

To measure the runtime of the protocol, we made a proof-of-concept implementation in C++. The implementation simulates all parties within a single process thread, thus performing all operations sequentially. All cryptographic operations use a key length of 2048 bits, and real values such as model weights are represented as 16-bit fixed-point numbers. Furthermore, data packing is used to speed up model updates.

The runtime tests were executed on a mobile workstation running Arch Linux on an Intel® Core™ i7-3610QM 2.3 GHz quad-core processor with 8 GB RAM. Figure 6.1 shows

the impact of the number of DSPs, and thus the total number of campaigns, on the total computation time. The time spent in the profile update protocol increases linearly with the number of DSPs since a profile update must be processed by each DSP. In a realistic setting, DSPs would operate in parallel, resulting in constant-time performance of the profile update protocol. The bidding and auction protocols cannot be fully parallelized due to the operations performed by the PSP and ad exchange, and thus scale linearly in the number of DSP. Since the model update protocol is only performed once for every advertisement shown, its runtime is independent of the number of DSPs.

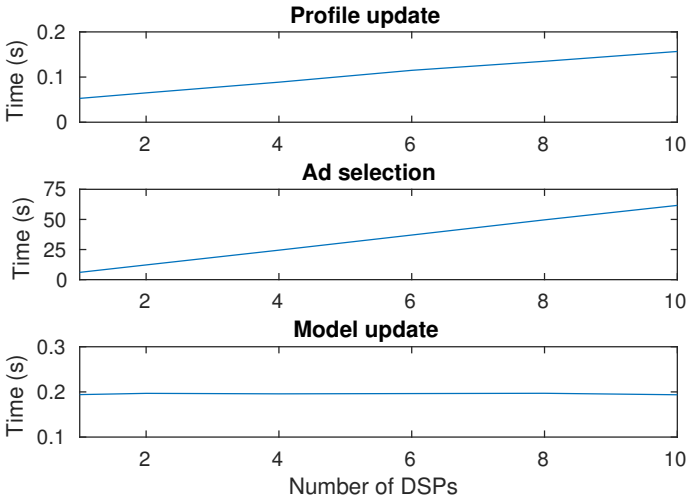


Figure 6.1: Total computation time required by a single run of each subprotocol for an increasing number of DSPs, with a fixed profile size $d = 2^{12}$ and number of campaigns per DSP $\kappa = 10$. Model update time is averaged using $\zeta = 10$.

Figure 6.2 shows the impact of profile dimensionality on the total computation time. It is apparent that, although all subprotocols scale linearly with profile size, high-dimensional profiles as used in practice result in dozens of seconds of computation time for every shown advertisement if not parallelized.

6.1.4. CONCLUSION AND FUTURE WORK

In this paper we present, to the best of our knowledge, the first protocol using machine learning over encrypted data to preserve privacy in OBA. DSPs must work together with a semi-honest, non-colluding PSP to estimate a user's response and the corresponding bid price within the encrypted domain, after which the PSP and ad exchange run a privacy-preserving auction to select the winning bid. Encrypted reports of ad clicks or views are aggregated by the PSP for billing and model update purposes. At no point are the contents of the user profile or the shown advertisement revealed to any party other than the user themselves, nor are model parameters revealed to any party other than the DSP. Finally, individual bid prices are revealed to no party at all.

The computation time required by AHEad quickly increases with profile dimension-

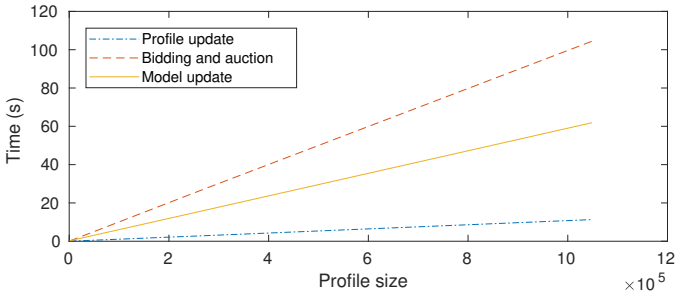


Figure 6.2: Total computation time required by a single run of each subprotocol for an increasing profile size d, with a single DSP running a single campaign. Model update time is averaged using $\zeta = 10$.

ality, resulting in over 100 seconds of computation per bid for $d = 2^{20}$ on our modest hardware. Moreover, an encrypted user profile of size 2^{20} requires 4 Gib of storage space and communication bandwidth between DSPs and the PSP. Nevertheless, our work shows promising initial results in terms of user privacy and achieves performance linear in the profile size, and calls for future research to provide further performance improvements. We are confident that, combined with sufficiently powerful hardware, the resulting protocols could spark a revolution in user privacy in OBA.

6.2. BADASS: PRESERVING PRIVACY IN BEHAVIOURAL ADVERTISING WITH APPLIED SECRET SHARING

Online advertising is a pervasive phenomenon on the Internet, forming one of the driving economic factors behind free web services. With a global spend of $178 billion in 2016 [1], online advertising forms a primary financial pillar supporting free web content by allowing publishers to offer content to users free of charge [2]. In recent years, however, an increasing number of people object to advertisements being shown on web pages they visit, resulting in a sharp increase in the use of technological measures to block advertisements. According to a 2017 report, an estimated 615 million devices have ad blocking tools installed, amounting to 11% of the Internet population, and the use of such tools is expected to grow further in the future [3]. The consequence of the increased use of ad blockers is that publishers experience a significant loss of revenue from the advertising space they offer. The global loss of advertising revenue due to ad blocking was estimated to be $41.4 billion in 2016, or 23% of the total ad spend [4]. Some publishers request that users disable ad blockers on their web pages, or deny access to ad blocker users altogether, in an effort to limit revenue loss due to ad blocking [6]. The consequence of such practices is an arms race between ad blocking technologies, and circumvention of ad blockers. These developments pose a threat to the business models of many free web services, necessitating measures to alleviate the objections against online advertising in order to attain a sustainable advertisement-supported Internet economy.

One of the objections people have against online advertising is the widespread data collection by advertising companies, which infringes on user privacy [5]. In a recent

survey among users of an ad blocking tool, 32% of respondents indicated that privacy concerns were among the reasons for their use of an ad blocker [6]. A similar survey on privacy and advertising showed that 94% of respondents considered online privacy an important issue, and 70% of respondents indicated that online advertising networks and online advertisers should be responsible for users' online privacy [7]. The widespread collection of user information that raises privacy concerns is a key element of behavioural targeting, in which a user's browsing behaviour determines which advertisements are shown to the user. Although such behavioural advertising is recognized as being beneficial to both users and publishers, a mistrust of advertising companies and a lack of control hinders acceptance of behavioural advertising [5].

ONLINE BEHAVIOURAL ADVERTISING

Online advertising is forming one of the driving economic factors behind free web services. With a global spend of $178 billion in 2016 [1], online advertising forms a primary financial pillar supporting free web content by allowing publishers to offer content to users free of charge [2]. In recent years, however, an increasing number of people object to advertisements being shown on web pages they visit, resulting in a sharp increase in the use of technological measures to block advertisements. According to a 2017 report, an estimated 615 million devices have ad blocking tools installed, amounting to 11% of the Internet population, and the use of such tools is expected to grow further in the future [3]. The consequence of the increased use of ad blockers is that publishers experience a significant loss of revenue from the advertising space they offer. The global loss of advertising revenue due to ad blocking was estimated to be $41.4 billion in 2016, or 23% of the total ad spend [4]. Some publishers request that users disable ad blockers on their web pages, or deny access to ad blocker users altogether, in an effort to limit revenue loss due to ad blocking [6]. The consequence of such practices is an arms race between ad blocking technologies, and circumvention of ad blockers. These developments pose a threat to the business models of many free web services, necessitating measures to alleviate the objections against online advertising in order to attain a sustainable advertisement-supported Internet economy.

A major concern for the users is their privacy which is threatened by the widespread data collection of advertising companies [5]. The collected data is used in behavioural targeting to determine which advertisements are shown to a user based on the user's browsing behaviour. Although such behavioural advertising is recognized as being beneficial to both users and publishers, a mistrust of advertising companies and a lack of control hinders acceptance of behavioural advertising [5]. In a recent survey among users of an ad blocking tool, 32% of respondents indicated that privacy concerns were among the reasons for their use of an ad blocker [6]. A similar survey on privacy and advertising showed that 94% of respondents considered online privacy an important issue, and 70% of respondents indicated that online advertising networks and online advertisers should be responsible for users' online privacy [7].

The practice of showing advertisements based on previously exhibited behaviour is known as OBA. In OBA, user interests are inferred from data such as visited web pages, search queries, and online purchases. Based on these user interests, advertisements are typically personalized using campaign-specific supervised machine learning models that predict users' responses to advertisements. Behavioural advertising greatly im-

proves the expected advertising effectiveness by targeting advertisements at individual users [8]. Users benefit from OBA by being served more relevant advertisements, and advertisers can reach a specific desired audience by using accurate targeting. Moreover, publishers benefit from an increased value of their advertising space.

OBA utilises the RTB model of buying and selling advertisements [9]. RTB facilitates real-time auctions of advertising space through marketplaces called ad exchanges (AdX), allowing buyers to determine bid values for individual ad impressions. The real-time nature of such auctions, in which bids are to be placed in a fraction of a second, allows fine-grained control over allocation of advertising budgets, but also requires the whole auction process to be carried out programmatically. Along with ad exchanges, other types of platforms emerged to manage the added complexity of RTB. DSPs provide advertisers, who may not possess the expertise required to accurately estimate impression values, with technologies to bid on individual impressions from multiple inventories. Likewise, SSPs support publishers in optimizing advertising yield.

In existing literature, a number of methods is proposed to address privacy concerns in OBA. These methods include blocking advertisements altogether [10], obfuscating browsing behaviour [11], and anonymization [12], as well as exposing only generalized user profiles to advertising companies [13]. Limiting the data that is available to advertising companies, however, is expected to decrease the targeting accuracy [14], and thus the value of advertisements to users, advertisers, and publishers. Other work proposes cryptographic approaches to aggregate click statistics [13] or select advertisements using secure hardware [15]. These approaches, however, are based on advertising models in which centralized networks perform simple keyword-based advertisement selection, and as such are unsuitable for use within the highly distributed RTB model. Recently, Helsloot et al. [29] proposed a protocol that uses threshold homomorphic encryption to preserve privacy in OBA within the RTB model. However, the use of expensive cryptographic operations throughout the protocol results in prohibitively large computational costs.

In this paper, we present BAdASS, a novel privacy-preserving protocol for OBA that is compatible with the RTB mechanism of buying ads and supports behavioural targeting based on highly detailed user profiles. BAdASS achieves significant performance improvements over the state of the art, using machine learning on secret-shared data to preserve privacy in OBA tasks. Our protocol uses the highly fragmented nature of the OBA landscape to distribute trust between parties, such that no single party can obtain sensitive information. We achieve performance multilinear in the size of user profiles and the number of DSP s, and perform the highly time-sensitive advertisement selection task in a fraction of a second.

In the rest of the paper, we summarize the preliminary methods in Section 6.2.1. In Section 6.2.2 we explain BAdASS in detail. In Section 6.2.3 we provide the performance analyses for our protocol based on communication and computation complexity and real-time experiments. We analyze the security of BAdASS in Section 6.2.4 and, we conclude the paper in Section 6.2.5.

6.2.1. PRELIMINARIES

LOGISTIC REGRESSION

Logistic regression is one possible technique for user response estimation which has been commonly used by advertising companies such as Google [16], Facebook [17], and Criteo [18]. Given a d-dimensional feature vector $\boldsymbol{x}$ and model parameters $\boldsymbol{w}$, it estimates the probability of a binary outcome using the sigmoid function:

$$\hat{y} = \sigma(\boldsymbol{w}^\top \boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{w}^\top \boldsymbol{x}}}. \tag{6.2}$$

In the concept of OBA, $\boldsymbol{x}$ contains the user profile while the binary output y indicates click or no click. Although a variety of machine learning algorithms has been proposed for user response estimation tasks, logistic regression has recently been used in production systems by many advertising companies, such as Google [16], Facebook [17], and Criteo [18], and is considered a state-of-the-art model for response prediction [19]. A logistic regression model is used to estimate the probability of a binary outcome y (click or no click) based on a d-dimensional predictor vector $\boldsymbol{x}$ (the user profile) [9]. Given a feature vector $\boldsymbol{x}$ and model parameters $\boldsymbol{w}$, the model predicts the click probability $\hat{y}$ using the sigmoid function: The model parameters are updated as $\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \boldsymbol{g}$ using the gradient of the logistic loss $\boldsymbol{g} = (\hat{y} - y)\boldsymbol{x}$ as in [16]. η in the update function is the learning rate which can be a constant or can be set to decay per iteration and per coordinate of the gradient vector. After observing the actual binary click label y, we use the gradient of the logistic loss to update model parameters $\boldsymbol{w}$ using an online SGD method as in [16]. Given the gradient of the loss $\boldsymbol{g} = (\hat{y} - y)\boldsymbol{x}$, we perform the model update $\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \boldsymbol{g}$, where η is the learning rate. Note that the learning rate can be a constant, but may also be set to decay per iteration and per coordinate of the gradient vector.

FEATURE HASHING

The feature space in logistic regression may become too large when categorical features with high cardinality are used. To avoid a high dimensional user vector, we use the hashing trick in [20] which enables to map the user profile into a lower-dimensional vector $\boldsymbol{x}$ by setting x_i to a count of the values whose hash is i. Many of the features considered in OBA are categorical. The high cardinality of some of these features results in a very large feature space, particularly if feature conjunctions are used. To encode categories from a user profile into a vector of fixed dimensionality for use in logistic regression, we use the hashing trick [20], which showed promising results in e.g. [18]. The hashing trick maps values from our high-dimensional user profile into a lower-dimensional vector $\boldsymbol{x}$ by setting x_i to a count of the values whose hash is i. The resulting d-dimensional vector $\boldsymbol{x}$ (in [18], $d = 2^{24}$ is used) is the input feature vector to the logistic regression model.

SHAMIR SECRET SHARING

Shamir's secret sharing scheme [30] is a t-out-of-n threshold scheme in which a secret $s \in \mathbb{Z}_p$ for a prime p is shared among n parties, from which any subset of size at least t can reconstruct the secret. We use the notation $\langle s \rangle$ to indicate a (t, n) Shamir secret sharing of a value s, for some predefined t and n, and $\langle \boldsymbol{v} \rangle$ denotes an element-wise Shamir sharing of the vector $\boldsymbol{v}$. Shamir's secret sharing scheme is additively homomorphic, such that

$\langle s_1 \rangle + \langle s_2 \rangle = \langle s_1 + s_2 \rangle$. Parties holding shares of two secret values can thus compute shares of the sum of the two values, without interaction with other parties. Furthermore, a public value c can be added to a shared secret s without interaction by adding c to each of the shares, i. e. $\langle s \rangle + c = \langle s + c \rangle$. Likewise, a shared secret can be multiplied with a public value c by multiplying each of the shares with c, i. e. $\langle s \rangle \cdot c = \langle cs \rangle$.

Multiplication of the shares of two secret values s_1 and s_2 results in a $(2(t-1), n)$ sharing of $s_1 \cdot s_2$, thus requiring $2t-1$ shares to reconstruct the secret. Ben-Or et al. [31] describe a multiplication protocol in which the resulting polynomial is reduced to degree $t-1$ and randomized. Given a sharing of a value s, where $\langle s \rangle_i$ is held by party i, the degree reduction step is performed by each party splitting their share $\langle s \rangle_i$ into a new (t, n) sharing $\langle s \rangle_{i,1}, \ldots, \langle s \rangle_{i,n}$. Each party i distributes their subshares among all parties, such that party j is given the subshare $\langle s \rangle_{i,j}$. Each party j then combines the received subshares $\langle s \rangle_{1,j}, \ldots, \langle s \rangle_{n,j}$ into a new share $\langle s \rangle'_j$. The resulting sharing $\langle s \rangle'$ is a new (t, n) sharing of the value s. Gennaro et al. [32] simplify the degree reduction and randomization steps into a single step, requiring a single round per multiplication. Note that n needs to be at least $2t-1$ for degree reduction to work.

UNIVERSAL RE-ENCRYPTION

Universal re-encryption, presented as a technique for mix networks by Golle et al. [33], allows re-randomization of a ciphertext without access to the public key that was used during encryption. In BAdASS, we use the universal re-encryption technique presented in [33], based on a multiplicatively homomorphic cryptosystem such as ElGamal [34]. We use the notation $[[x]]_u$ to denote the encryption of a value x under the public key of user u using a universal re-encryption scheme.

6.2.2. PROTOCOL DESIGN

In designing a privacy-preserving online advertising system, we aim to satisfy three goals. The first goal is to ensure profile privacy, such that information from which the interests of a user can be inferred is not revealed to any party other than the user. Moreover, we aim to ensure model privacy, such that model parameters used by a bidder are not revealed to any party other than the bidder. Finally, the system must be applicable to the RTB model and integrated into the online advertising landscape, allowing bidders to estimate the value of an ad impression based on historic observations.

Our protocol is based on a semi-honest security model. Since some existing companies act as both AdX and DSP, we assume that the AdX colludes with DSPs. To assist in privacy-preserving computations in the presence of colluding parties, we introduce an additional entity into our setting called PSP. The PSP is not trusted with private data in unencrypted form, but is assumed to follow the protocol specification without colluding with any other party. We assume that targeting is performed only by DSPs, such that DSPs are the only parties that operate on user data. The AdX only collect bids, and from these bids select the winner. SSPs only offer advertising space to multiple ad exchanges, and are not considered in our protocol.

Parties collaboratively compute bid values based on a logistic regression model using secret-shared user profiles and model parameters. We define a DSP group Γ_i to be a set of DSPs of size at least $m = 2t-1$ for a given reconstruction threshold t. Computations on

behalf of a DSP $\gamma_{i,j} \in \Gamma_i$ are performed entirely within Γ_i. In our protocol descriptions, any operations performed by DSPs on secret-shared values are assumed to be performed by all DSPs in a DSP group Γ_i. Plaintext values and encrypted values are generated by the DSP responsible for the campaign on which computations are performed and, where necessary, published within Γ_i.

BAdASS is divided into four different phases: user profiling, bidding, auction, and model update. Prior to protocol execution, advertisers set up campaigns such that DSPs can bid on their behalf, and the PSP splits DSPs into groups of at least m parties. Moreover, each DSP shares campaign-specific parameters among the DSPs in their group. Finally, each user generates a key pair using any multiplicatively homomorphic asymmetric cryptosystem and publishes their public key. In the following subsections, we explain the four phases of BAdASS. A summary of symbols used in the description of the protocol is provided in Table 6.4.

Table 6.4: Explanation of symbols used in BAdASS

Symbol	Explanation
u	Unique user identifier.
v	Unique bid request identifier.
k	Unique campaign identifier.
Γ_i	DSP group i.
$\gamma_{i,j}$	Unique DSP identifier, where $\gamma_{i,j}$ is the j^{th} DSP of DSP group Γ_i
K_{Γ_i}	Set of campaigns run by DSP group Γ_i.
$\boldsymbol{x}$	Input feature vector obtained by feature hashing.
$\boldsymbol{w}_k$	Model parameter vector for a campaign k, where $w_{k,i}$ is the i^{th} coordinate of $\boldsymbol{w}_k$.
$\boldsymbol{c}_k$	Bidding function parameters for a campaign k.
η_k	Learning rate parameter for campaign k.
b_k	Bid value for campaign k.
a_k	Advertisement associated with campaign k.
$\pi(\boldsymbol{x})$	Random permutation function, which re-orders the elements of vector $\boldsymbol{x}$.
$\pi^{-1}(\cdot)$	Inverse permutation of $\pi(\cdot)$, such that $\pi^{-1}(\pi(\boldsymbol{x})) = \boldsymbol{x}$.
M	List of vectors containing information associated with bid values for use in the auction protocol.

USER PROFILING PHASE

To preserve privacy during the user profiling phase, browsing behaviour is recorded locally within the user's web browser using an existing technique such as RePriv [23]. The

resulting profile is captured in a d-dimensional feature vector $\boldsymbol{x}$ using feature hashing. To reduce the communication costs associated with sending the full d-dimensional feature vector for each request, feature vectors are cached at DSPs. Caching allows periodic background updates of feature vectors, minimizing delays experienced by the user during the time-sensitive advertisement selection phase. To securely share a feature vector among DSPs without knowing the topology of the OBA landscape, the user splits their profile into two additive shares, one of which is given to the AdX, the other to the PSP. Both the AdX and the PSP, in turn, create Shamir shares from their additive shares, which are distributed among the DSP groups for which the profile update is intended. Every DSP within the group thus receives two Shamir shares, one from each additive share created by the user, which are combined into a single Shamir share of the original value by calculating the sum of the two shares. The user profiling phase is illustrated in Algorithm 19.

Algorithm 19 Profile update protocol, executed jointly between a user, AdX, PSP and DSP group, and initiated periodically by every user.

procedure USER:SEND-PROFILE-SHARE($\boldsymbol{x}, u$)
  Pick $\boldsymbol{r} \in_R \mathbb{Z}_p^d$
  $\langle\langle \boldsymbol{x} \rangle\rangle_1 \leftarrow \boldsymbol{x} - \boldsymbol{r}$
  $\langle\langle \boldsymbol{x} \rangle\rangle_2 \leftarrow \boldsymbol{r}$
  invoke SHARE-PROFILE($u, \langle\langle \boldsymbol{x} \rangle\rangle_1$) at AdX
  invoke SHARE-PROFILE($u, \langle\langle \boldsymbol{x} \rangle\rangle_2$) at PSP

procedure SHARE-PROFILE($u, \langle\langle \boldsymbol{x} \rangle\rangle_m$)
  $\langle\langle\langle \boldsymbol{x} \rangle\rangle_m\rangle \leftarrow$ SHAMIR-SHARE($\langle\langle \boldsymbol{x} \rangle\rangle_m$)
  for all $\gamma_{i,j} \in \Gamma_i$ **do**
    invoke DSP:COMBINE-PROFILE($u, \langle\langle\langle \boldsymbol{x} \rangle\rangle_m\rangle_{\gamma_{i,j}}$) at DSP $\gamma_{i,j}$

procedure DSP:COMBINE-PROFILE($u, \langle\langle\langle \boldsymbol{x} \rangle\rangle_m\rangle$)
  store $\langle\langle\langle \boldsymbol{x} \rangle\rangle_m\rangle$ for user u
  if $\langle\langle\langle \boldsymbol{x} \rangle\rangle_1\rangle$ and $\langle\langle\langle \boldsymbol{x} \rangle\rangle_2\rangle$ are both stored **then**
    $\langle \boldsymbol{x}_u \rangle \leftarrow \langle\langle\langle \boldsymbol{x} \rangle\rangle_1\rangle + \langle\langle\langle \boldsymbol{x} \rangle\rangle_2\rangle$

Depending on the feature hashing method used, the feature vector $\boldsymbol{x}$ is either binary or contains only small values. Assuming the use of a binary feature vector, it would be ideal from the user's perspective to create additive shares of the feature vector in $\mathbb{Z}_2$, rather than $\mathbb{Z}_p$, as smaller shares reduce the required communication bandwidth. Subsequent computations on the user profile, however, must be performed in $\mathbb{Z}_p$ to represent real values with sufficient precision. In our setting with two additive shares, securely converting shares in $\mathbb{Z}_2$ to shares in $\mathbb{Z}_p$ requires an invocation of the multiplication protocol for every feature vector dimension, resulting in high computation and communication costs for DSPs. We therefore favour sharing the user profile in $\mathbb{Z}_p$.

BIDDING PHASE

The bidding phase starts when a users contacts an AdX with an ad request. Receiving the ad request, AdX sends a bid request to DSP groups each of which cooperatively calcu-

lates the bidding prices for the campaigns they are responsible for. For each campaign, the user response $\hat{y}$ is estimated using a logistic regression model, and bidding values are derived from response estimations using linear bidding functions $B(\hat{y}) = c_1\hat{y} + c_2$ for campaign-specific constants c_1 and c_2. A challenge in logistic regression is to compute sigmoid function within the secret-shared domain. In existing literature the sigmoid function is computed either by approximation [24, 25] or in clear [26, 27, 29]. In this work, we let the PSP compute the sigmoid function in the clear, as approximation leads to a degradation of predictive performance and incurs additional computational costs. The input to the sigmoid function $\boldsymbol{w}^\intercal \boldsymbol{x}$ is thus revealed to the PSP. In our setting, this is acceptable as the PSP knows neither the user, nor the campaign a value is associated with. Therefore, the PSP cannot infer any more information than that there exists a user who is interested in a topic. Moreover, a DSP group could submit additional randomly generated values to mask real inputs.

During the bidding phase, every DSP group cooperatively calculates bidding prices for each of the campaigns the group is responsible for. When a user contacts an AdX with an ad request, the AdX sends a bid request to every DSP group, each of which executes the bidding protocol.

The bidding protocol is outlined in Algorithm 20. The model parameters $\boldsymbol{w}_k$ for campaigns $k \in K_{\Gamma_i}$, where K_{Γ_i} is the set of campaigns run within Γ_i, and the user profile $\boldsymbol{x}_u$ of a user u, are shared within Γ_i. The multiplications that are required for the calculation of the inner product of $\boldsymbol{w}_k$ and $\boldsymbol{x}_u$ in line 3 are performed locally, without degree reduction. Since the results of these local multiplications are not used in further multiplications, the sum of all multiplied values is a single sharing $\langle s_k \rangle$ of degree $2t-2$. As the PSP subsequently collects and combines all $m \geq 2t-1$ shares of s_k, no degree reduction step is required in calculating $\boldsymbol{w}^\intercal \boldsymbol{x}$.

Since campaign parameters $c_{k,1}$ and $c_{k,2}$ are private to the DSP responsible for campaign k, they are secret-shared among the parties in the DSP group. Calculation of the bid price $b_k = c_{k,1}\hat{y}_k + c_{k,2}$ therefore requires a single invocation of the multiplication protocol for every campaign, which can be parallelized such that all bid values are calculated in a single round of communication. To ensure profile privacy, each advertisement a_k is encrypted using the user's public key. The encrypted advertisement is submitted to the PSP, via the AdX such that the PSP cannot link the submission to a specific DSP, along with a random number r_k and the group descriptor Γ_i. Finally, the PSP stores a mapping $r_k \rightarrow ([[a_k]]_u, \Gamma_i)$, which is used in the auction phase to retrieve the advertisement.

AUCTION PHASE

The auction protocol uses a hierarchical auction in which each DSP group engages in a secure comparison protocol to select the highest of the bids within the DSP group, along with associated information that is used in the model update phase. Shares of the information associated with the highest bid are stored for later use, after which each DSP group submits their highest bid to a global auction to select the final winner. Note that, due to the use of secret sharing, the global auction cannot be performed by the AdX alone. In order to maintain the same level of trust as in the bidding protocol, at least m parties are required in the auction protocol. Therefore, the global auction is not performed by the ad exchange, but by a randomly selected DSP group Γ^*.

Algorithm 20 Bidding protocol, executed jointly by a DSP group Γ_i and the PSP, and invoked by the AdX for a user u at every DSP group.

1: **procedure** DSP:CALCULATE-BID($\{\langle \boldsymbol{w_k} \rangle, \langle \boldsymbol{c}_k \rangle \mid k \in K_{\Gamma_i}\}, u$)
2: **for all** $k \in K_{\Gamma_i}$ **do**
3: $\langle s_k \rangle \leftarrow \sum_{i=1}^{d} \langle x_{u,i} \rangle \cdot \langle w_{k,i} \rangle$
4: Pick a unique random value r_k
5: Store mapping $r_k \rightarrow ([[a_k]]_u, \Gamma_i)$ at PSP via AdX
6: Pick random permutation function $\pi(\cdot)$
7: $\langle \boldsymbol{s}' \rangle \leftarrow \pi(\langle \boldsymbol{s} \rangle)$
8: **invoke** $\langle \hat{\boldsymbol{y}}' \rangle \leftarrow$ PSP:CALCULATE-SIGMA($\langle \boldsymbol{s}' \rangle$) at PSP
9: $\langle \hat{\boldsymbol{y}} \rangle \leftarrow \pi^{-1}(\langle \hat{\boldsymbol{y}}' \rangle)$
10: **for all** $k \in K_{\Gamma_i}$ **do**
11: $\langle b_k \rangle \leftarrow \langle c_{k,1} \rangle \cdot \langle \hat{y}_k \rangle + \langle c_{k,2} \rangle$

12: **procedure** PSP:CALCULATE-SIGMA($\langle \boldsymbol{s} \rangle$)
13: $\boldsymbol{s} \leftarrow$ combine $\langle \boldsymbol{s} \rangle$
14: **for all** $s_i \in \boldsymbol{s}$ **do**
15: $\hat{y}_i \leftarrow \sigma(s_i)$
16: **return** $\langle \hat{\boldsymbol{y}} \rangle$

The auction protocol is shown in Algorithm 21. The protocol relies on a secure comparison protocol that takes as input shares of two values a and b, and gives as output shares of 1 if $a \geq b$, and shares of 0 otherwise. Such a protocol is described by e. g. Reistad and Toft [35]. During the procedure to find the maximum bid, shares of the highest bid and additional information associated with the highest bid are obtained via multiplication with the result of the comparison. After the global comparison, shares of a random identifier r associated with the highest bid are sent to the PSP, where the shares are combined to retrieve the encrypted advertisement and group descriptor associated with the highest bid. To ensure unlinkability between the encrypted advertisement retrieved from the PSP after the auction and the values submitted prior to the auction, the PSP performs re-randomization of the encrypted advertisement on line 23 using universal re-encryption. Finally, the encrypted ad and the group descriptor, as well as the bid request identifier v, are sent via the AdX to the user, who decrypts and displays the advertisement.

MODEL UPDATE PHASE

During the model update phase, the response prediction model associated with the shown advertisement is updated using the update rule from Section 6.2.1. In order to ensure unlinkability between users and campaigns, the model update protocol is split into three stages. During the first stage, the user identifier is revealed to the DSP group responsible for the shown advertisement in order to calculate shares of the update gradient $\boldsymbol{g} = \eta(\hat{y} - y)\boldsymbol{x}$. In the second stage, each DSP submits a set of multiple gradient shares to the PSP, which mixes the received shares via random rotation. The PSP then re-

Algorithm 21 Auction protocol, executed jointly by every DSP group Γ_i, an auction group Γ^*, and the PSP.

procedure DSP:PREPARE-AUCTION($v, \langle \boldsymbol{b} \rangle, \langle \boldsymbol{r} \rangle, \langle \hat{\boldsymbol{y}} \rangle, \langle \boldsymbol{\eta} \rangle, \langle \boldsymbol{k} \rangle$)
  for all Γ_i **do**
    $\langle M_i \rangle \leftarrow \left(\langle \boldsymbol{r_i} \rangle, \langle \hat{\boldsymbol{y}}_{\boldsymbol{i}} \rangle, \langle \boldsymbol{\eta_i} \rangle, \langle \boldsymbol{k_i} \rangle \right)$
    $\left(\langle b_i^{max} \rangle, \langle r_i^{max} \rangle, \langle \hat{y}_i^{max} \rangle, \langle \eta_i^{max} \rangle, \langle k_i^{max} \rangle \right) \leftarrow$ MAX-BID($\langle \boldsymbol{b_i} \rangle, \langle M_i \rangle$)
    Store mapping $v \rightarrow (\langle b_i^{max} \rangle, \langle \hat{y}_i^{max} \rangle, \langle \eta_i^{max} \rangle, \langle k_i^{max} \rangle)$
  invoke PERFORM-AUCTION($v, \langle \boldsymbol{b^{max}} \rangle, \langle \boldsymbol{r^{max}} \rangle$) at Γ^*

procedure PERFORM-AUCTION($v, \langle \boldsymbol{b} \rangle, \langle \boldsymbol{r} \rangle$)
  $(\perp, \langle r^{max} \rangle) \leftarrow$ MAX-BID($\langle \boldsymbol{b} \rangle, \langle \boldsymbol{r} \rangle$)
  invoke PSP:SEND-AD($\langle r^{max} \rangle$) at PSP

procedure MAX-BID($\langle \boldsymbol{b} \rangle, \langle M \rangle$)
  $\langle \hat{b} \rangle \leftarrow \langle b_1 \rangle$
  for $j \leftarrow 1, [\langle M \rangle]$ **do**
    $\langle \hat{M}_j \rangle \leftarrow \langle M_{j,1} \rangle$
  for $i \leftarrow 2, [\langle \boldsymbol{b} \rangle]$ **do**
    $\langle \rho \rangle \leftarrow \langle b_i \rangle \geq \langle \hat{b} \rangle$
    $\langle \hat{b} \rangle \leftarrow \langle \rho \rangle \cdot \langle b_i \rangle + (1 - \langle \rho \rangle) \cdot \langle \hat{b} \rangle$
    for $j \leftarrow 1, [\langle M \rangle]$ **do**
      $\langle \hat{M}_j \rangle \leftarrow \langle \rho \rangle \cdot \langle M_{j,i} \rangle + (1 - \langle \rho \rangle) \cdot \langle \hat{M}_j \rangle$
  return $\langle \hat{b} \rangle, \langle \hat{M} \rangle$

procedure PSP:SEND-AD($\langle r \rangle$)
  $r \leftarrow$ combine $\langle r \rangle$
  $([[a]]_u, \Gamma_i) \leftarrow$ lookup r
  Re-randomize $[[a]]_u$
  Send $([[a]]_u, \Gamma_i)$ to user via AdX

shares the set of gradient shares among the DSP group. In the final stage, the campaign identifiers of the set of gradients are revealed to the DSP group, allowing the DSP group to apply the gradients calculated in the first stage to the correct parameter vector. Since the gradient shares have been mixed, the DSP group cannot link values revealed in the third phase to values revealed in the first phase. The model update protocol, described in detail in Algorithm 22, is initiated by a user u, who reports shares of their response y directly to the responsible DSP group based on the group descriptor Γ_i received along with the advertisement.

In the first phase, each DSP in Γ_i calculates shares of $\delta = \eta(\hat{y} - y)$, which is multiplied with each element of the user profile to form $\boldsymbol{g}$. These multiplications are performed locally, without reducing the degree of the result. The update gradient shares, bid value shares, and campaign identifier shares are locally stored as lists $\langle G \rangle$, $\langle B \rangle$, and $\langle K \rangle$ until sufficient values are aggregated for mixing. When sufficient values are accumulated,

Algorithm 22 Model update protocol, invoked by the user at the DSP group responsible for the displayed advertisement.

procedure DSP:PREPARE-MODEL-UPDATE($v, u, \Gamma_i, \langle y \rangle$)
  $\left(\langle b \rangle, \langle \hat{y} \rangle, \langle \eta \rangle, \langle k \rangle\right) \leftarrow$ lookup v
  $\langle \delta \rangle \leftarrow$ TRUNCATE($\langle \eta \rangle \cdot (\langle \hat{y} \rangle - \langle y \rangle)$)
  for $i \leftarrow 1, d$ **do**
    $\langle g_i \rangle \leftarrow \langle x_{u,i} \rangle \cdot \langle \delta \rangle$
  $(\langle G \rangle, \langle B \rangle, \langle K \rangle) \leftarrow \left(\langle G \rangle \cup \langle \boldsymbol{g} \rangle, \langle B \rangle \cup \langle b \rangle, \langle K \rangle \cup \langle k \rangle\right)$
  if sufficient values are accumulated **then**
    for all $\gamma_{i,j} \in \Gamma_i$ **do**
      Pick random $r_{i,j}$
      Rotate $\left(\langle G \rangle_{i,j}, \langle B \rangle_{i,j}, \langle K \rangle_{i,j}\right)$ $r_{i,j}$ times
    invoke PSP:MIX-SHARES($\Gamma_i, \langle G \rangle, \langle B \rangle, \langle K \rangle$) at PSP

procedure PSP:MIX-SHARES($\Gamma_i, \langle G \rangle, \langle B \rangle, \langle K \rangle$)
  if shares from all $\gamma_{i,j} \in \Gamma_i$ have been received **then**
    Rotate $(\langle G \rangle, \langle B \rangle, \langle K \rangle)$ by a random value
    Re-share $(\langle G \rangle, \langle B \rangle, \langle K \rangle)$ as $\left(\langle G' \rangle, \langle B' \rangle, \langle K' \rangle\right)$
    invoke UPDATE-MODEL($\langle G' \rangle, \langle B' \rangle, \langle K' \rangle$) at Γ_i

procedure DSP:UPDATE-MODEL($\langle G' \rangle, \langle B' \rangle, \langle K' \rangle$)
  for all $\gamma_{i,j} \in \Gamma_i$ **do**
    Rotate $\left(\langle G' \rangle_{i,j}, \langle B' \rangle_{i,j}, \langle K' \rangle_{i,j}\right)$ back $r_{i,j}$ times
  $K \leftarrow$ combine $\langle K' \rangle$
  for all $\langle \boldsymbol{g}' \rangle \in \langle G' \rangle$, $\langle b' \rangle \in \langle B' \rangle$, $k \in K$ **do**
    for $i \leftarrow 1, d$ **do**
      $\langle w_{k,i} \rangle \leftarrow \langle w_{k,i} \rangle - \langle g'_i \rangle$
    $\langle \tilde{b}_k \rangle \leftarrow \langle \tilde{b}_k \rangle + \langle b' \rangle$

each DSP sends its shares to the PSP for mixing. To prevent recombination of shares by the PSP, each DSP $\gamma_{i,j}$ rotates their lists of shares by a random number $r_{i,j}$. Given a sufficiently large aggregation threshold, the average number of attempts needed for the PSP to successfully combine the received shares becomes prohibitively large. The PSP subsequently picks a random number r_{PSP}, and rotates each of the lists of shares r_{PSP} times, such that the positions of output values cannot be linked to the positions of input values.

The share values themselves need also be randomized before being sent back to the DSPs to prevent linking input values to output values. Since the PSP does not know which received shares belong to the same value due to the rotation by DSPs, randomization cannot be performed by adding shares of zero. Instead, the PSP splits each received share $\langle s \rangle_i$ into a new sharing $\langle s \rangle_{i,1}, \ldots, \langle s \rangle_{i,n}$. These subshares are distributed among the DSPs in Γ_i in a manner analogous to the degree reduction step described in 6.2.1, such that each party j receives subshares $\langle s \rangle_{1,j}, \ldots, \langle s \rangle_{n,j}$. The DSPs then recombine the received subshares to obtain new shares of the original values $\langle s \rangle_i$. To match subshares originating

from the same value, each DSP rotates the subshares originating from DSP $\gamma_{i,j}$ back $r_{i,j}$ times. After recombining the rotated subshares, each DSP has lists $\langle G' \rangle$, $\langle B' \rangle$, and $\langle K' \rangle$, where each of the lists contains shares of the same values as were submitted to the PSP, rotated r_{PSP} times.

The DSPs combine $\langle K' \rangle$ to reveal the list of campaign identifiers to which the gradient and bid shares belong. Due to the mixing of the lists, DSPs cannot link the campaign identifiers revealed in the third phase to user identifiers revealed in the first phase, provided a sufficiently large mixing threshold is chosen. Finally, each DSP locally updates their shares of the model parameter vectors with the list of gradient shares, and adds the received bid shares to the bid value aggregates.

6.2.3. PERFORMANCE ANALYSIS

To evaluate the performance of BAdASS, we provide both a theoretical analysis of the computational and communication complexities of the subprotocols, and a set of measurements obtained from a proof-of-concept implementation. The theoretical analysis provides an overview of the performance impact incurred by the use of secret sharing for each of the parties, whereas measurements from the implementation show the actual performance of the protocol, taking into account all required computations.

COMPUTATIONAL COMPLEXITY

The computational complexity of BAdASS depends on a number of variables, in particular the user profile dimensionality d, the number of campaigns K, and the update aggregation threshold ζ. The used variables are summarized in Table 6.5.

In the profile update protocol, the user creates d additive sharings, and the AdX and PSP both create d Shamir sharings. Moreover, each DSP performs d additions. If the profile update protocol is invoked for all DSP groups at once, the computational complexity is therefore $O(dn)$, where n is the total number of DSPs. In the bidding protocol, each DSP performs a multiplication for every campaign within its DSP group to calculate the bid value, and an encryption of the advertisement for all its own campaigns. Calculation of shares of the inner product $\langle \boldsymbol{w}^\intercal \boldsymbol{x} \rangle$ is performed locally, as explained in Section 6.2.2, and since the resulting value is reconstructed by the PSP, no degree reduction step is necessary. The multiplications involved in calculating the inner product are thus 'free'. In the auction protocol, each DSP group Γ_i performs $K_i - 1$ comparisons, where K_i is the number of campaigns of Γ_i, followed by a single DSP group Γ^* performing $g-1$ comparisons, where g is the number of groups. Since the group Γ^* is chosen at random out of g groups for every auction, the amortized complexity of the the auction phase is $O(K)$. In the model update protocol, the d multiplications to compute the update gradient need no degree reduction step because the shares are mixed at the PSP, and are thus 'free'. The mixing step is performed for a batch of ζ updates once every ζ invocations, resulting in an amortized cost equal to that of processing a single update. To process a single update, the PSP re-shares all m shares of the d-dimensional update gradients, where m is the size of DSP groups, followed by the DSP performing d local recombinations of the created subshares. The total cost of the re-sharing, in terms of sharing and reconstructing secrets, is equal to that of dm multiplications. The group size m, however, can be considered a constant determined by the recombination threshold, resulting in an amortized com-

Table 6.5: Symbols used in the computational analysis of BAdASS.

Symbol	Description
d	Dimensionality of user profiles.
n	Number of DSPs.
g	Number of DSP groups.
m	Size of a DSP group.
κ	Number of campaigns of a DSP.
K	Total number of campaigns.
K_i	Number of campaigns within a DSP group Γ_i.
ζ	Number of model updates accumulated per DSP group.
σ	Size in bits of a secret share.
ξ	Size in bits of an advertisement tuple, consisting of an encrypted advertisement, a group descriptor, and a random identifier.
λ	Total number of comparisons required to find the maximum bid. Equal to $\lceil \log_2 (K_i - 1) \rceil + \lceil \log_2 (g - 1) \rceil$.
ρ	Number of rounds required by a single run of the comparison protocol.
γ	Number of bits transmitted in a single run of the comparison protocol.
T	Number of rounds required by a single run of the truncation protocol.
τ	Number of bits transmitted in a single run of the truncation protocol.

plexity of $O(d)$ for the model update phase.

COMMUNICATION COMPLEXITY

Table 6.6 lists the amortized number of bits transmitted by each party for each subprotocol, as well as the number of rounds of communication required by each subprotocol. The round complexities of the profile update and bidding protocols are constant. Since the group size m is bounded by a constant, and the share size σ is constant, the communication complexity of the profile update phase can be considered linear with respect to the profile size d. Note that if the user profile is distributed among multiple DSP groups, the complexity of the user profiling phase becomes multilinear in the profile size and the number of DSPs. During the bidding phase, the AdX acts as a proxy to transmit a total of K advertisement tuples, and the PSP transmits K sharings of the estimated user response. Each DSP performs K_i multiplications, each requiring m shares to be transmitted, sends K_i shares of inner products to the PSP, and sends an advertisement tuple for each of its κ own campaigns to the PSP via the AdX. The total communication complexity of the bidding phase is thus $O(K)$, or linear in the number of campaigns.

The auction protocol consists of λ comparisons, where λ is logarithmic with respect to the number of campaigns within a group and the number of groups. The round complexity of the auction phase is thus $O(\log_2 K)$, or logarithmic in the number of cam-

Table 6.6: Communication bandwidth in bits and number of rounds of communication per invocation of each subprotocol of BAdASS. γ and τ denote the number of bits transferred in the comparison and truncation protocols. ρ is the round complexity of the comparison protocol, and T is the round complexity of the truncation protocol.

Protocol	Rounds	User	AdX	DSP	PSP
Profiling	2	$2d\sigma$	$dm\sigma$		$dm\sigma$
Bidding	2		$K\xi$	$(m+1)K_i\sigma + \kappa\xi$	$Km\sigma$
Auction	$\lambda(\rho+1)+3$		ξ	$(5K_i-3)m\sigma + K_i\gamma + 2\frac{1}{g}\sigma$	ξ
Update	$T+1\frac{3}{\zeta}$	$m\sigma$		$(d+1)m\sigma + \tau + (d+3)\sigma$	$(d+2)m^2\sigma$

paigns, provided the round complexity of the comparison protocol is constant. Since each DSP performs an average of K_i comparisons per invocation of the auction protocol and transmits a fixed number of shares for each multiplication, the communication complexity of the auction protocol is linear in the number of campaigns, provided the communication complexity of the comparison protocol is constant.

The model update protocol contains a single invocation of the truncation protocol, of which the number of rounds is considered constant, as well as one round of multiplication. Every ζ invocations, three more rounds for mixing and combining are performed. The amortized round complexity of the model update protocol is therefore constant. Although the amount of bits transmitted by the PSP contains a factor m^2, we can assume this to be a small constant due to the small upper bound on m. The average communication complexity of the model update protocol is $O(d)$, provided the communication complexity of the truncation protocol is constant and the group size m is bounded by a constant.

IMPLEMENTATION

To measure the runtime of BAdASS, we made a proof-of-concept implementation of the protocol in C++. The secure comparison protocol is based on the protocol by Reistad and Toft [35] as implemented in **VIFF**[1] [36]. Real values, such as model weights, are represented as 16-bit fixed-point numbers. All operations using Shamir shares are performed in a prime field of order $p = 2^{31} - 1$, such that share values can be represented as 32-bit integers. The reconstruction threshold t is set to 3, resulting in a DSP group size m of 5. The key length for the ElGamal cryptosystem is set to 2048 bits to achieve a sufficiently high security level[2].

The setup used for runtime measurements is similar to that used in [29]. The tests were executed on a mobile workstation running Arch Linux on an Intel® Core™ i7-3610QM 2.3 GHz quad-core processor with 8 GB RAM. Similar to [29], all parties are simulated within a single process thread, thus performing all operations sequentially. Figure 6.3 shows a comparison between the runtimes of BAdASS and the state-of-the-art

[1] **VIFF**. Available online at `https://github.com/kljensen/viff/blob/master/viff/comparison.py`

[2] See e. g. `https://www.keylength.com` for key lengths as recommended by various organizations. The NIST considers a key length of 2048 sufficiently secure until 2030.

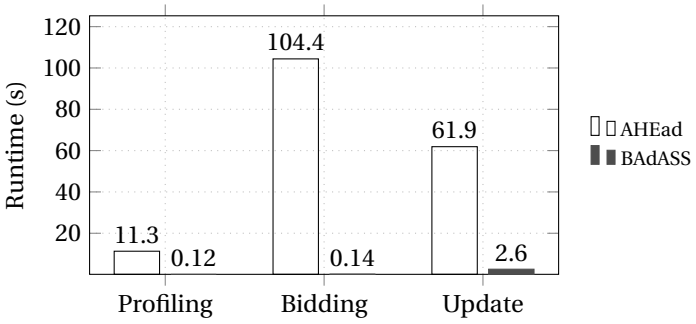


Figure 6.3: Performance comparison between BAdASS and the state-of-the-art AHEad protocol. The runtimes are measured in similar settings, using profile dimensionality $d = 2^{20}$ for both protocols. Note that the runtime measurements for AHEad are performed using a single DSP running a single campaign, whereas 5 DSPs with a total of 5 campaigns are simulated on a single process thread for BAdASS due to the recombination threshold.

AHEad protocol for the different protocol phases. The comparison makes it evident that, for a realistically large profile size $d = 2^{20}$, BAdASS provides significant performance improvements over AHEad for every subprotocol, with the time-sensitive bidding phase requiring less than 150 ms for a DSP group. The computation time required by the model update protocol of BAdASS far exceeds that of the profile update, bidding, and auction protocols, due to the large number of subshare recombinations performed by the DSPs as well as the large number of sharings created by the PSP. Note that the computation time of the model update protocol is averaged, since the expensive mixing and update steps are only performed once every $\zeta = 10$ invocations of the protocol.

Table 6.7: Runtime measurements for each step of each model update for BAdASS in ms. Mix shares and update model steps are executed once every $\zeta = 10$ invocations while the update preparation step is performed for every viewed ad. The total shows the average of ζ invocations.

Protocol	**DSP**	**PSP**
Prepare update	16.87	—
Mix shares	—	2399.67
Update model	23406.70	—
Total	2597.51	

Specifically, as shown in Table 6.7, one invocation of these steps is more expensive than the preparation step by a factor of 140 and 1400, respectively. The cost of performing the model update protocol when the computation time is averaged becomes

$$\frac{10 \times 16.87 + 2399.67 + 23406.70}{10} = 2597.51 \text{ ms}, \tag{6.3}$$

which is approximately 2.6 seconds, as shown in Figure 6.3. Based on our measurements, we estimate that if the computations performed by the DSPs are parallelized, the average time spent on the model update protocol for a profile dimensionality of $d = 2^{20}$ drops from 2.6 seconds to about 750 ms per invocation. The computation times of the bidding and auction phase and the profile update phase are both below 150 ms for large profile sizes, even when DSPs are simulated sequentially, and thus seem very well suited for use in a real-time setting as required by the RTB advertising model. The relatively large amount of computation performed in the model update phase is less time sensitive, and can thus be periodically performed as a background task without harming the user experience.

In Table 6.8, we list the average communication bandwidth of our protocol for specific parameters used in our implementation. We use a share size $\sigma = 32$ bits, and a group size of $m = 5$ DSPs. Each DSP is responsible for $\kappa = 10$ campaigns, and with two groups the total number of campaigns is $K = 100$. The profile dimensionality is $d = 2^{20}$, and the size of an advertisement descriptor is assumed to be 4160 bits, of which 4096 bits are the encrypted advertisement, 32 bits the random identifier, and 32 bits the group descriptor. From the table, it is evident that the profile update and model update require a significant amount of communication, with up to 100 MiB per invocation, on average, of the model update protocol. The time-sensitive bidding and auction protocols, however, require very little bandwidth. Moreover, very little bandwidth is used by the user, with only the periodically executed profile update protocol requiring more than a few dozen bytes at 8 MiB per invocation, making the bandwidth use of the user very acceptable for modern unmetered connections.

Table 6.8: Bandwidth usage of BAdASS in KiB per invocation of each subprotocol, based on realistic parameters used in our implementation of BAdASS.

Protocol	Rounds	User	AdX	DSP	PSP
Profiling	2	8192	20480	0	20480
Bidding	2	—	51	6.25	2
Auction	$\lambda(\rho+1)+3$	—	0.5	$5+K_i\gamma$	0.5
Update	$T+1\frac{3}{\zeta}$	0.02	—	$24576+\tau$	102400

6.2.4. SECURITY OF BADASS

The security requirements of BAdASS are satisfied by the security of the underlying secret-sharing and encryption schemes in the semi-honest setting. In the non-interactive phases of the protocol, both the user profile and model parameters are shared among a DSP group using Shamir's secret sharing scheme, which provides information-theoretic security as long as no more than $t-1$ parties collude.

In the profile update protocol, the user profile is shared between the PSP and the AdX using a two-party additive secret sharing scheme, which, given the assumption that the PSP does not collude with any party, provides information-theoretic security. The additive shares are split into Shamir shares before being sent to a DSP group, where the

additive shares are combined into a single Shamir share. Since the PSP and the AdX only receive additive shares, and DSPs obtain a single Shamir share of each additive share, the PSP, AdX and DSPs gain no knowledge of the contents of user profiles.

In the bidding protocol, the PSP obtains values $\boldsymbol{w}_k^\intercal \boldsymbol{x}_u$ and $\hat{\boldsymbol{y}}_k$ from DSP groups, but does not know the campaign k or user u to which the values belong, nor the specific DSP responsible for the campaign. Since the PSP knows neither $\boldsymbol{w}_k$ nor $\boldsymbol{x}_u$, inferring the individual values of $\boldsymbol{w}_k$ or $\boldsymbol{x}_u$ from $\boldsymbol{w}_k^\intercal \boldsymbol{x}_u$ is equivalent to the hardness of solving the subset-sum problem. Given a set of positive integers $S = \{a_1, a_2, \cdots a_n\}$ and an integer b, the subset-sum problem aims to find whether there exist a subset of S, for which the summation equals to b [37]. Finding such a subset, however, is an NP-complete problem. In BAdASS, $\boldsymbol{x}_u$ is a vector with binary or small values, and $\boldsymbol{w}_k$ contains 16-bit fixed-point numbers. Assuming a binary $\boldsymbol{x}_u$ the multiplication $\boldsymbol{w}_k^\intercal \boldsymbol{x}_u$ is actually a selection of indices of $\boldsymbol{w}_k$ based on the value in every index of $\boldsymbol{x}_u$. Finding a subset of $\boldsymbol{w}_k$, where the sum of the subset is equal to $\boldsymbol{w}_k^\intercal \boldsymbol{x}_u$ is hard, when $\boldsymbol{w}_k$ and $\boldsymbol{x}_u$ are private and both of them have size $d = 2^{20}$.

If the PSP receives multiple values of $\boldsymbol{w}_k^\intercal \boldsymbol{x}_u$ for the same $\boldsymbol{w}_k$ and $\boldsymbol{x}_u$, the PSP can link these values to the same user, but cannot learn any information about the user's interests as the PSP cannot link response predictions to campaigns. The PSP also receives a mapping between a randomly generated number and an advertisement encrypted using the universal re-encryption scheme, which is semantically secure under the DDH assumption [33]. The use of universal re-encryption provides key privacy, such that the PSP does not learn the identity of the user, and the randomization of ciphertexts in the ElGamal cryptosystem ensures that different submissions of the same advertisement cannot be linked. During the auction protocol, the PSP learns the random number associated with the winning bid in order to retrieve the winning advertisement, but since the mappings are anonymized by the AdX, the PSP cannot link this value to a DSP.

In the model update protocol, the PSP obtains rotated shares of update gradients, bid values, and campaign identifiers. Given unbounded computational power, the PSP can perform an exhaustive search of rotation coefficients until recombination of shares results in likely values. Choosing sufficiently large values for the update period ζ and recombination threshold t makes exhaustive searches infeasible. After the PSP mixes the shares, DSPs receive shares of the same values submitted earlier in the model update phase. Since the shares are re-shared by the PSP, however, DSPs cannot link the shares received after mixing to shares submitted before mixing. Moreover, the random rotation performed by the PSP prevents DSPs from linking inputs to outputs.

6.2.5. CONCLUSION

In this paper we present a novel protocol using machine learning over secret-shared data to preserve privacy in OBA with minimal user-noticeable delays. Trust is distributed among DSPs using threshold secret sharing, allowing DSPs to collaboratively compute bid prices and determine the highest bid without gaining any knowledge of a user's interests. Reports of individual clicks and views of advertisements are secret-shared among DSPs, where they are used to privately update model parameters via a mixing step at the PSP. At no point are the contents of user profiles, shown advertisements, and actual user responses revealed to any party other than the user, nor are model parameters revealed

to any party other than the DSP responsible for the campaign. Individual bid prices are not revealed to any party, but are aggregated for billing purposes. Finally, the protocols are integrated into the RTB setting by forming DSP groups from existing DSPs, with the addition of a single new party.

BAdASS achieves significant performance improvements over previous work. AHEad protocol presented in [29] requires more than 100 seconds of computation time to calculate a single bid value in the time-sensitive bidding phase. In comparison, BAdASS simulates the calculation of 5 bid values in less than 150 milliseconds in a similar setup, even without parallelization across DSPs. BAdASS is even efficient enough to serve advertisements in real time as required by the RTB model, provided the communication between DSPs incurs minimal latency. Despite the overhead of the model update protocol, the results obtained with BAdASS show that by applying secret sharing techniques a level of performance can be achieved in the time-sensitive bidding and auction phases that does not degrade the perceived responsiveness.

To the best of our knowledge, BAdASS is the first protocol to allow sub-second behavioural targeting of advertisements while preserving user privacy. The heavily fragmented shape of the online advertising landscape lends itself particularly well to the use of efficient secret-sharing techniques, giving advertising companies the opportunity to cooperatively move towards acceptable forms of behavioural advertising. Although the presented protocol should be adapted to the malicious setting, as DSPs may have an incentive to modify competitors' bid values, the results obtained with BAdASS show that it is possible to serve behaviourally targeted advertisements without disclosing those interests to any party, all within a fraction of a second. We believe that these results provide a first step towards adoption of privacy-preserving methods in the online advertising ecosystem.

REFERENCES

[1] V. Letang and L. Stillman, *Global Advertising Forecast*, Tech. Rep. (MAGNA, 2016).

[2] J. Deighton and H. M. Brierley, *Economic Value of the Advertising-Supported Internet Ecosystem*, Tech. Rep. (Interactive Advertising Bureau, 2012).

[3] PageFair, *The State of the Blocked Web*, Tech. Rep. (2017).

[4] PageFair and Adobe, *The Cost of Ad Blocking*, Tech. Rep. (2015).

[5] B. Ur, P. G. Leon, L. F. Cranor, R. Shay, and Y. Wang, *Smart, useful, scary, creepy: perceptions of online behavioral advertising,* in *Symposium On Usable Privacy and Security, SOUPS '12, Washington, DC, USA, July 11-13, 2012* (ACM, 2012) pp. 4:1–4:15.

[6] M. An, *Why People Block Ads (And What It Means for Marketers and Advertisers)*, Tech. Rep. (HubSpot Research, 2016).

[7] TRUSTe, *2011 Consumer Research Results: Privacy and Online Behavioural Advertising*, Tech. Rep. (2011).

[8] J. Yan, N. Liu, G. Wang, W. Zhang, Y. Jiang, and Z. Chen, *How much can behavioral targeting help online advertising?* in *Proc. of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009* (ACM, 2009) pp. 261–270.

[9] J. Wang, W. Zhang, and S. Yuan, *Display advertising with real-time bidding (RTB) and behavioural targeting,* CoRR **abs/1610.03013** (2016), arXiv:1610.03013 .

[10] G. Merzdovnik, M. Huber, D. Buhov, N. Nikiforakis, S. Neuner, M. Schmiedecker, and E. R. Weippl, *Block me if you can: A large-scale study of tracker-blocking tools,* in *2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, Paris, France, April 26-28, 2017* (IEEE, 2017) pp. 319–333.

[11] M. Degeling and T. Herrmann, *Your interests according to google - A profile-centered analysis for obfuscation of online tracking profiles,* CoRR **abs/1601.06371** (2016), arXiv:1601.06371 .

[12] F. Papaodyssefs, C. Iordanou, J. Blackburn, N. Laoutaris, and K. Papagiannaki, *Web identity translator: Behavioral advertising and identity privacy with WIT,* in *Proc. of the 14th ACM Workshop on Hot Topics in Networks, Philadelphia, PA, USA, November 16 - 17, 2015* (ACM, 2015) pp. 3:1–3:7.

[13] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas, *Adnostic: Privacy preserving targeted advertising,* in *Proc. of the Network and Distributed System Security Symposium, NDSS 2010, San Diego, California, USA, February 28 - March 3, 2010* (The Internet Society, 2010).

[14] J. Estrada-Jiménez, J. Parra-Arnau, A. Rodríguez-Hoyos, and J. Forné, *Online advertising: Analysis of privacy threats and protection approaches,* Computer Communications **100**, 32 (2017).

[15] M. Backes, A. Kate, M. Maffei, and K. Pecina, *Obliviad: Provably secure and practical online behavioral advertising,* in *IEEE Symposium on Security and Privacy, SP 2012,San Francisco, California, USA, 21-23 May 2012* (IEEE Computer Society, 2012) pp. 257–271.

[16] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, S. Chikkerur, D. Liu, M. Wattenberg, A. M. Hrafnkelsson, T. Boulos, and J. Kubica, *Ad click prediction: a view from the trenches,* in *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013* (ACM, 2013) pp. 1222–1230.

[17] X. He, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, A. Atallah, R. Herbrich, S. Bowers, and J. Q. Candela, *Practical lessons from predicting clicks on ads at facebook,* in *Proc. of the 8th International Workshop on Data Mining for Online Advertising, ADKDD 2014, New York City, New York, USA, August 24, 2014* (ACM, 2014) pp. 5:1–5:9.

[18] O. Chapelle, E. Manavoglu, and R. Rosales, *Simple and scalable response prediction for display advertising,* ACM TIST **5**, 61:1 (2014).

[19] A. Szwabe, P. Misiorek, and M. Ciesielczyk, *Logistic regression setup for RTB CTR estimation,* in *Proc. of the 9th International Conference on Machine Learning and Computing, Singapore, Singapore*, ICMLC 2017 (ACM, 2017) pp. 61–70.

[20] K. Q. Weinberger, A. Dasgupta, J. Langford, A. J. Smola, and J. Attenberg, *Feature hashing for large scale multitask learning,* in *Proc. of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, ACM International Conference Proceeding Series, Vol. 382 (ACM, 2009) pp. 1113–1120.

[21] P. Paillier, *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes,* in *Advances in Cryptology — EUROCRYPT '99* (1999) pp. 223–238.

[22] C. Hazay, G. L. Mikkelsen, T. Rabin, T. Toft, and A. A. Nicolosi, *Efficient RSA key generation and threshold paillier in the two-party setting,* J. Cryptology **32**, 265 (2019).

[23] M. Fredrikson and B. Livshits, *Repriv: Re-imagining content personalization and in-browser privacy,* in *32nd IEEE Symposium on Security and Privacy, S&P 2011, Berkeley, California, USA, 22-25 May 2011* (IEEE Computer Society, 2011) pp. 131–146.

[24] Q. Zhang, L. T. Yang, and Z. Chen, *Privacy preserving deep computation model on cloud for big data feature learning,* IEEE Trans. Computers **65**, 1351 (2016).

[25] T. Chen and S. Zhong, *Privacy-preserving backpropagation neural network learning,* IEEE Trans. Neural Networks **20**, 1554 (2009).

[26] C. Orlandi, A. Piva, and M. Barni, *Oblivious neural network computing via homomorphic encryption,* EURASIP J. Information Security **2007**, 18:1 (2007).

[27] Y. Aono, T. Hayashi, L. T. Phong, and L. Wang, *Scalable and secure logistic regression via homomorphic encryption,* in *Proc. of the 6th ACM Conference on Data and Application Security and Privacy, CODASPY 2016, New Orleans, LA, USA, March 9-11, 2016* (ACM, 2016) pp. 142–144.

[28] M. Nateghizad, Z. Erkin, and R. L. Lagendijk, *An efficient privacy-preserving comparison protocol in smart metering systems,* **2016**, 11.

[29] L. J. Helsloot, G. Tillem, and Z. Erkin, *AHEad: Privacy-preserving online behavioural advertising using homomorphic encryption,* in *IEEE International Workshop on Information Forensics and Security, WIFS 2017, Rennes, France, December 4-7, 2017* (IEEE, 2017) pp. 1–6.

[30] A. Shamir, *How to share a secret,* Commun. ACM **22**, 612 (1979).

[31] M. Ben-Or, S. Goldwasser, and A. Wigderson, *Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract),* in *Proc. of the 20th Annual ACM Symposium on Theory of Computing, Chicago, Illinois, USA, May 2-4, 1988* (ACM, 1988) pp. 1–10.

[32] R. Gennaro, M. O. Rabin, and T. Rabin, *Simplified VSS and fact-track multiparty computations with applications to threshold cryptography,* in *Proc. of the 17th Annual ACM Symposium on Principles of Distributed Computing, PODC '98, Puerto Vallarta, Mexico, June 28 - July 2, 1998* (ACM, 1998) pp. 101–111.

[33] P. Golle, M. Jakobsson, A. Juels, and P. F. Syverson, *Universal re-encryption for mixnets,* in *Topics in Cryptology - CT-RSA 2004, The Cryptographers' Track at the RSA Conference 2004, San Francisco, CA, USA, February 23-27,2004*, Lecture Notes in Computer Science, Vol. 2964 (Springer, 2004) pp. 163–178.

[34] T. E. Gamal, *A public key cryptosystem and a signature scheme based on discrete logarithms,* IEEE Trans. Information Theory **31**, 469 (1985).

[35] T. I. Reistad and T. Toft, *Secret sharing comparison by transformation and rotation,* in *Information Theoretic Security - Second International Conference, ICITS 2007, Madrid, Spain, May 25-29, 2007, Revised Selected Papers*, Lecture Notes in Computer Science, Vol. 4883 (Springer, 2007) pp. 169–180.

[36] M. J. B. Geisler, *Cryptographic Protocols: Theory and Implementation*, Ph.D. thesis, Department of Computer Science, Aarhus University (2010).

[37] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, 3rd Edition* (MIT Press, Cambridge, Massachusetts - London, England, 2009).

7

DISCUSSION

Outsourced data analytics benefit small- and medium-sized companies that do not have sufficient computational resources and knowledge in performing data analytics by enabling an external party, a service provider, to perform the analytics for these companies. However, leaking sensitive information through the outsourced data or computation results is a major concern for the analytics outsourcing companies. Considering the legal and financial outcomes of data breaches, the analytics outsourcing companies desire an outsourced data analytics service that can assure the protection of their sensitive data and results.

In this thesis, we investigated the confidentiality concerns in data analytics-as-a-service. We defined several specifications that assure confidentiality in data analytics-as-a-service, which are input & output privacy, accuracy, and efficiency. We selected three applications of data analytics-as-a-service whose privacy concerns have not been investigated thoroughly in the existing literature, namely process analytics, machine learning, and marketing analytics. We provided privacy-preserving protocol designs for each analytics type that aim to assure the confidentiality specifications. Our protocols focus on the secure computation of data analytics. Thus, we used two cryptographic techniques for the protection and processing of sensitive data, which are homomorphic encryption and secure multiparty computation. Below we summarize the proposed protocols for each analytics type.

PROCESS ANALYTICS

Among the three steps of process analytics defined in Chapter 2, in this thesis, we focused on the design of privacy-preserving protocols for process discovery and conformance checking. In Chapter 3, we proposed two protocols for the discovery of processes from encrypted data using homomorphic properties of encryption. Both of our protocols based on a client-server scenario, where the client helps the server for the intermediate steps of computation.

Our first protocol, AlphaSec, transforms an existing process discovery algorithm, the alpha algorithm, to a protected domain. In the design of AlphaSec, we used two-party

protocols that are based on homomorphic encryption. To the best of our knowledge, our protocol is the first protocol that performs process discovery under data protection, but it has several limitations. One limitation is the practicability of the alpha algorithm, which cannot handle noise and incompleteness. Another limitation of the protocol is its efficiency. The two-party protocols based on homomorphic encryption incurs a significant computation cost on the client's side, which may not be affordable by every analytics outsourcing company.

Considering the limitations of AlphaSec, we proposed a second protocol, PriSM, which generalizes our first protocol to more robust process discovery algorithms. Instead of limiting our solution to a specific discovery algorithm, our protocol aims to discover directly-follows graphs that can be used as a basis for the well-known process discovery algorithms. Furthermore, to get rid of the high computation cost of the two-party homomorphic encryption-based protocols, we proposed to use two variants of ElGamal encryption, i.e. additively homomorphic and multiplicatively homomorphic, such that the computations switch from multiplicative variant to additive variant. With a similar bandwidth usage of the first protocol, in PriSM, we can improve the computation performance of the process discovery under encryption.

In Chapter 4, we proposed two protocols for conformance checking considering the privacy-efficiency trade-off in computations. We proposed our protocols in a two-server setting such that the two servers share the computation tasks among them. We used secure two-party computation, more specifically arithmetic sharing and Boolean sharing, to protect and process the sensitive data. Both of our protocols aim to compute the optimal alignment between a log trace and a process model by solving the shortest path problem. Our first protocol, $\text{SCORCH}_{\text{EXH}}$, uses an exhaustive approach that computes all possible paths to find the optimal alignment. While it guarantees complete privacy protection, for larger process models or longer traces the solution can be inefficient. As an alternative, our second protocol, $\text{SCORCH}_{\text{PQ}}$, solves the shortest path problem using a private priority queue. The solution based on the private priority queue might leak information about the length of the output, however, the computation time is significantly less compared to $\text{SCORCH}_{\text{EXH}}$.

MACHINE LEARNING

In Chapter 5, we proposed SwaNN, a protocol for private neural network predictions that focuses on convolutional neural networks. Our protocol can work in two different scenarios: a client-server scenario and a non-colluding server scenario. In a non-colluding server scenario, our solution can perform predictions for two images simultaneously. We used a hybrid mechanism for the protection and processing of sensitive data such that the computations switch between homomorphic encryption and secure two-party computation. We used homomorphic encryption for the computation of arithmetic operations such as additions or scalar multiplications. For the non-linear operations, such as comparisons, we used arithmetic and Boolean circuits. The experimental results show that our solution can provide a good balance for the trade-off between computation cost and bandwidth usage. While balancing the computation and communication cost, our solution uses well-known cryptographic techniques that enable the reproducibility of the computation tasks conveniently.

MARKETING ANALYTICS

In Chapter 6, we proposed two protocols for online behavioral advertising which are used as a part of marketing analytics. Our protocols are in a non-colluding servers scenario such that a semi-trusted party, privacy service provider (PSP), collaborates with data service providers (DSPs) or ad exchanges to perform computations without colluding. Both of our protocols combine machine learning with secure computation techniques to match the right advertisement for a given user profile. Our first protocol, AHEad, uses an additively homomorphic cryptosystem to protect and process sensitive user data. To distribute the trust between parties it uses a threshold variant of the homomorphic cryptosystem. Due to the usage of homomorphic encryption with large ciphertext sizes, the computations can be significantly expensive. Therefore, as an alternative, we propose BAdASS, a protocol based on secret-sharing in a multiparty setting. Since in secret-sharing the expansion in the message size is not significant, we can achieve a significant improvement in computation cost compared to AHEad.

7.1. ACHIEVEMENTS

Our review of the existing literature in Chapter 2 shows that the research efforts that aim to meet the confidentiality specifications for the selected applications are limited. To respond to the inadequacy of privacy-preserving protocols for the selected analytics applications, in this thesis, we propose protocols that aim to achieve input/output privacy, accuracy, and efficiency in confidential DAaaS. We observed that we are able to achieve input and output privacy with the selection of provably secure cryptographic techniques in our protocols. However, achieving efficiency and accuracy in secure computation is not trivial. Thus, in the design of the protocols, our focus was to assure accuracy and efficiency as stated in the following research question:

> *Which cryptographic techniques and optimization methods can be used to improve the computation and communication performance in confidential data analytics-as-a-service while maximizing the accuracy of algorithms?*

In the design of our protocols, we use two main cryptographic techniques which are partially homomorphic encryption and secure multiparty computation. In our designs, we saw that with the repetitive usage of these techniques, the cost of computation and communication increases drastically. Therefore, we use several optimization techniques, such as data packing or multi-exponentiation. In the following, we clarify how the techniques we use in our designs address our research question by elaborating on the subquestions designed in Section 1.4.

- *How efficiently can a service provider company perform process analytics in confidential DAaaS, where the accuracy of process analytics algorithms are maintained?*

  We deal with the efficiency of computations in process analytics in two different types of process mining by using different cryptographic techniques. In Chapter 3, we use homomorphic encryption to perform process discovery. We propose two protocols, such that the first protocol AlphaSec uses the additively homomorphic Paillier cryptosystem [1], whereas the second protocol PriSM uses the ElGamal

cryptosystem [2] with the additive and the multiplicative homomorphic variants. In AlphaSec, we are able to handle homomorphic additions efficiently. However, for other types of operations, such as multiplications or equality checks, we use two-party protocols that require some intermediate decryption. To eliminate the computation overhead of the two-party protocols, we use data packing that reduces the computation and communication cost incurred by the intermediate decryptions. In PriSM, on the other hand, we change the protocol design such that the operations can be performed using only homomorphic additions and multiplications, which are efficiently performed with the homomorphic properties of the ElGamal cryptosystem. The only computation bottleneck in PriSM is the switching phase from the multiplicatively homomorphic variant to the additively homomorphic variant, where data packing operations are not feasible.

In Chapter 4, we perform conformance checking under privacy preservation using arithmetic circuits and Boolean circuits [3] in a two-party setting. We choose secure two-party computation over homomorphic encryption since the computations in conformance checking require the repetitive amount of equality checks and comparisons, which can be performed efficiently using Boolean circuits compared to homomorphic encryption. In the design of our secure conformance checking protocols, we use arithmetic circuits to perform additions and multiplications. We switch the circuit type to Boolean type when a secure comparison or equality check is necessary. By reducing the usage of Boolean circuits to only non-linear operations, we reduce the computation time and bandwidth usage in our protocols significantly.

- *How can the cost of computation and communication be balanced by a service provider company who performs private neural network operations in confidential DAaaS?*

  In Chapter 5, we propose SwaNN for private neural network predictions which brings together two cryptographic techniques, additively homomorphic encryption and secure two-party computation with a switching phase. We observed that the convolutional and fully connected layers of neural networks can be computed efficiently using homomorphic encryption since the underlying operation is a dot product which can be computed with scalar multiplications and additions. On the other hand, the activation and pooling layers require computation of non-linear operations such as comparisons or multiplications. Thus, to compute these layers, we use arithmetic and Boolean circuits as implemented in [4] in a two-party setting.

  To reduce the cost of computations, we applied some optimization techniques to secure two-party computation and homomorphic encryption. We used single instruction multiple data operations (SIMD) [5] to simultaneously perform computations in secure two-party computation. To improve the computation performance of homomorphic encryption, we used two techniques which are data packing and multi-exponentiations. Data packing enables us to reduce the number of decryptions performed in switching from homomorphic encryption to secure two-party computation. On the other hand, using the multi-exponentiation tech-

nique in [6], we can perform the dot products in the convolutional layer and the fully connected layer simultaneously.

- *What is the feasibility of operating Real-Time Bidding mechanism for online behavioral advertising using cryptographic techniques?*

  In Chapter 6, we used additively homomorphic Paillier encryption and Shamir's secret sharing scheme [7] for online behavioral advertising. The nature of Real-Time Bidding mechanism requires to serve advertisements to users in real-time which cannot tolerate significant overhead on computation and communication cost. We observed that using additively homomorphic encryption we can perform the operations in a two-party setting, where PSP collaborates with a DSPs or an ad exchange without colluding. However, the computation overhead of the homomorphic encryption may not be tolerated in RTB setting, since the computation cost can exceed 100 seconds for a single bid. Instead, using secret sharing, we observed that we can reduce the computation time less than a second which is more suitable for the real-time setting.

  To improve the performance further, in our protocols, when it is possible, we chose to perform computations locally in plaintext. For instance, the computation of the sigmoid function for linear regression is delegated to PSP to prevent the loss in accuracy and the additional computation. However, it is important to remark that doing so does not affect the privacy of the protocol. The value provided to PSP is an aggregate and inferring the individual values requires to solve the subset sum problem. Furthermore, with the given information, PSP is not able to associate the value with a user or a specific ad campaign.

Above we discussed how our protocols achieve accuracy and efficiency for each selected application type specifically. In the rest of this section, we summarize how our designs handle the challenges related to accuracy and efficiency that are introduced in Section 2.4.

- **Loss of precision:** The mechanism we use in secure computation, i.e. homomorphic encryption and secure multiparty computation, work with integer values. Therefore, in our computations minimizing the accuracy loss due to a conversion from real values to integer values was an important task. In the computation of process analytics, the values used in process discovery and conformance checking are categorical values. Therefore, to be able to perform computations on these values we use a mapping function that maps the categorical values to integer values. Doing so, we were able to maintain the accuracy of our computations. Similarly, in online behavioral advertising, the user inputs provided to the protocols we categorical attributes and we used binary values to represent them. However, in neural network predictions the weight values and the result of activation functions are not necessarily integers. Thus, in Chapter 5, we used scaling on input values to maintain the precision of input values.

- **Nonlinear functions:** For the computation of nonlinear functions, we used three different approach in our protocols. The first approach is to use a polynomial approximation of the function as discussed in Chapter 5. However, we observed that

the approximation can reduce the accuracy of the computations. Thus, as an alternative we switched the mechanism that is used for secure computation. More specifically, knowing that we can perform more flexible operations using Boolean circuits, when necessary, we switched the computation from homomorphic encryption to secure two-party computation to perform non-linear operations. We observed that we can maintain the accuracy of computations with the switching mechanism while achieving feasible computation and communication cost with the use of several optimization techniques such as data packing and SIMD operations. The third approach that we use for nonlinear functions is to perform the function in plaintext as proposed in Chapter 6. This approach might not be secure if the data in plaintext leaks information about users. However, in our setting, the plaintext data is an aggregate value and inferring individual values from it requires to solve subset sum problem.

- **Trade-off between computation cost and bandwidth usage:** All of the protocols we propose require at least two parties to perform computations. Thus, minimizing both computation and communication cost is the major goal in our designs. As described above, we used several techniques such as data packing, SIMD, or multi-exponentiations to reduce both computation and communication cost.

7.2. REFLECTION

We showed that our proposals assure confidentiality in data analytics-as -a-service using cryptographic techniques and optimization methods to optimize the accuracy and efficiency. However, we are aware of the fact that the proposed techniques and scenarios can have limitations, and there can be alternative mechanisms. In this section, we elaborate on the limitations and discuss what could have done alternatively.

SELECTION OF THE SCENARIOS

In confidential DAaaS, we presented three possible scenarios to perform secure computations, which are a standalone server scenario, a client-server scenario, and a non-colluding servers scenario. Despite these scenarios provide certain advantage for the analytics outsourcing company to reduce computational cost on the client side, they can have several drawbacks. Below we discuss possible limitations of these scenarios.

- **Standalone server scenario:** This scenario is the desired scenario for the analytics outsourcing companies since it does not require a computation cost from their side. However, the computations performed in this scenario can be performed by a single party, i.e. service provider company, which might limit the flexibility of computations. In such a scenario, using homomorphic encryption is a reasonable choice for the protection and processing the data by a single party. As we discussed previously, using homomorphic encryption, a single party can perform only additions and/or limitations without decryption. Thus, the accuracy of computations is degraded.
- **Client-server:** As an alternative to the standalone server scenario, in this scenario, the computation tasks can be shared between a client and a server. Such a scenario enables the two parties to compute nonlinear operations since the client

(who hold the secret keys) can perform decryptions. However, in the distribution of computations, the workload on the client-side is important. Since the client does not have the same computational capabilities as the server, this scenario is feasible when the computation tasks on the client do not exceed the client's capacity.

- **Non-colluding servers:** This scenario enables two servers with similar computational resources to perform computations on the clients' behalf. It is a useful scenario by reducing the computational costs on the client-side while providing more accurate computation results compared to a standalone server scenario. However, it is suitable for the applications in which the collaboration of two or more service providers to perform analytics without colluding is feasible.

The protocol we propose in this thesis use either a client-server scenario or a non-colluding servers scenario. Since we want to maximize the accuracy in computations we did not prefer to use standalone server scenario. However, we believe that the solutions we propose can be applied in different scenarios as well. For instance, the solutions proposed in Chapter 3, which are in a client-server scenario, can be performed in a non-colluding servers scenario, using a threshold variant of chosen cryptosystems. Furthermore, the PriSM protocol in Chapter 3 can also be adapted to a standalone server scenario using a fully homomorphic cryptosystem since the required operations are only additions and multiplications. However, while using such a setting can eliminate the communication cost, the expensive nature of fully homomorphic encryption might increase the computation cost drastically.

SELECTION OF THE ADVERSARIAL BEHAVIOUR

All protocols we propose achieve security against semi-honest adversaries. In this setting, we assume the parties involved in the computations follow the tasks assigned to themselves without deviating from the protocol, but they are curious to get additional information from the intermediary messages and output they receive. We think that this is a realistic assumption in confidential data analytics-as-a-service since the companies are bonded with business contracts whose violation can have significant financial and legal consequences for the companies.

Designing protocols that are secure against malicious adversaries can provide better security guarantees. However, achieving security against malicious adversaries requires to use several additional techniques such as commitments or zero-knowledge proofs [8] whose implementations create overhead in computation and communication cost. Since the companies involved in DAaaS are bonded with business contracts, they are not expected to act maliciously. Thus, we believe assuring security against semi-honest adversaries is sufficient in confidential data analytics-as-a-service.

SELECTION OF THE ANALYTICS APPLICATIONS

In this thesis, we focus on three applications of data analytics-as-a-service, which are process analytics, machine learning, and marketing analytics. As stated previously, we observed that the existing literature that investigates the privacy concerns in these applications is not extensive. Therefore, with this thesis, we aim to contribute to research in confidential DAaaS with a focus on the selected analytics types.

However, the applications that we use in the thesis are also related to other types of data analytics applications. For instance, in process discovery, the challenge of mining directly sequential patterns is a subfield of frequent pattern mining algorithms. Thus, it can be related to several other techniques such as association rule mining [9] or sequential rule mining [10]. Similarly, the underlying problem in the computation of conformance checking with alignments is significantly related to the shortest path computation. Therefore, our contribution to secure conformance checking can be associated with graph mining algorithms that focus on shortest path problems. Our proposal on private neural network predictions focuses on convolutional neural networks. However, the proposed solutions can be also useful for other types of artificial networks. Finally, our proposal in Chapter 6 is based on logistic regression which is one of the well-known machine learning algorithms. The proposed solution can be adapted to other data analytics applications that make use of logistic regression.

7.3. FUTURE WORK

With this thesis, we contributed to confidential data analytics-as-a-service by designing protocols based on secure computation techniques. Our contributions are specific to three analytics applications which are process analytics, machine learning, and marketing analytics. We are aware of the fact that the protocols proposed in this thesis are open to improvements and extensions. Below we discuss future directions of research in confidential DAaaS.

We proposed the first protocols for process mining that achieves privacy preservation using provably secure cryptographic mechanisms. While our proposals are the first attempts in process discovery and conformance checking, we see that there is a need to extend the research in privacy-preserving process mining. For instance, our PriSM protocol in Chapter 3, which proposes a generic algorithm for process discovery, can be specialized to the commonly used process discovery algorithms. Similarly, our proposal for secure conformance checking can be extended with different cost functions. Furthermore, in this thesis, we have not focused on the third type of process mining which is process enhancement. A future direction of research can be to investigate the privacy challenges in process enhancement.

Regarding the privacy concerns in private neural network predictions, we see several possible directions for future research. One possible direction is to increase the accuracy of computations in predictions. Although the existing proposals for private neural network predictions can have a high accuracy rate, since the nonlinear functions such as sigmoid or hyperbolic tangent cannot be performed efficiently under data protection, the accuracy of predictions is still less than the accuracy of original computations. Thus, increasing the accuracy with a better design of non-linear functions requires further investigation. Another research direction is related to the efficiency and scalability of private neural network predictions which are currently limited to smaller neural networks and datasets. More efficient designs that can work on larger datasets are needed for more realistic neural network operations. Furthermore, investigating the privacy challenges in different neural network types can be another direction of future research.

Finally, achieving real-time computations in the online behavioral advertisements is an important challenge. With our proposals, we showed that the computation time can

be reduced significantly with the use of secret-sharing mechanisms. However, the heavy bandwidth usage in real-time bidding mechanism under privacy-preservation is still a bottleneck on the computations. The research on private online behavioral advertising can be extended with a focus on the improvement of communication performance.

REFERENCES

[1] P. Paillier, *Public-key cryptosystems based on composite degree residuosity classes,* in *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding* (1999) pp. 223–238.

[2] T. E. Gamal, *A public key cryptosystem and a signature scheme based on discrete logarithms,* IEEE Trans. Information Theory **31**, 469 (1985).

[3] O. Goldreich, S. Micali, and A. Wigderson, *How to play any mental game or A completeness theorem for protocols with honest majority,* in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA* (ACM, 1987) pp. 218–229.

[4] D. Demmler, T. Schneider, and M. Zohner, *ABY - A framework for efficient mixed-protocol secure two-party computation,* in *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015* (2015).

[5] N. P. Smart and F. Vercauteren, *Fully homomorphic SIMD operations,* Des. Codes Cryptography **71**, 57 (2014).

[6] C. H. Lim and P. J. Lee, *More flexible exponentiation with precomputation,* in *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings* (Springer, 1994) pp. 95–107.

[7] A. Shamir, *How to share a secret,* Commun. ACM **22**, 612 (1979).

[8] O. Goldreich, *The Foundations of Cryptography - Volume 1: Basic Techniques* (Cambridge University Press, 2001).

[9] J. Vaidya and C. Clifton, *Privacy preserving association rule mining in vertically partitioned data,* in *KDD* (ACM, 2002) pp. 639–644.

[10] C. C. Aggarwal, M. Bhuiyan, and M. A. Hasan, *Frequent pattern mining algorithms: A survey,* in *Frequent Pattern Mining* (Springer, 2014) pp. 19–64.

ACKNOWLEDGEMENTS

In the last pages of my thesis, I would like to express my gratitude to the people who were along with me on this journey and made it much more fun. Pursuing a Ph.D. itself is a challenge, and doing it in another country with no friends from home makes it even more difficult. However, I was lucky to meet great people that made this journey full of laughter, coffee, and sweet stuff.

First of all, I would like to express my gratitude to my promotor Prof. Inald Lagendijk for giving me the opportunity to pursue my Ph.D. under his supervision. Thank you Inald for your guidance, for the constructive feedback and discussions. It was a great pleasure for me to work with you.

My special thanks are extended to my daily supervisor Zeki Erkin. This was all possible thanks to you. Not only a supervisor, but you also became a brother and a friend to me. Thanks for believing in my potential and encouraging me in my ambitions. I am sad that I cannot have a morning coffee with you anymore, but I am happy that I got a lifelong friend.

I would like to thank my committee members Melek Önen, Mauro Conti, Geert-Jan Houben, Arie van Deursen, and Jeroen van den Hoven for taking time to review my dissertation and being a part of my defense ceremony. I am grateful for the support by Harm and Tommy in the translation of the propositions and the summary of my thesis.

I am thankful to my co-authors Mina Sheikhalishahi, Nicola Zannone, Tomas Keviczky, Rohan Hoogervorst, Yingqian Zhang, Sicco Verwer, Prahesa Setia, and Srinath Nandakumar for our collaboration and the members of the BSR project for the valuable discussions.

During my Ph.D. education, I got the chance of working in EURECOM as a guest researcher. I would like to thank Melek Önen for that. The three months that I spent in EURECOM was one of the best times of my life. I learned a lot about research, but I also enjoyed the beautiful summer of French Riviera. Thanks to you, I even had a chance to walk on the red carpet of the Cannes Film Festival. My EURECOM adventure would not be any fun without Beyza. I am so happy that we worked together and we explored everything together.

Ph.D. life is no fun without office mates, and Zeki's unique taste in choosing students makes it even better. Chibuike, thanks a lot for all the adventures in the first year of our Ph.D. studies, for arranging our trip to Nigeria, and for giving me the honor of being the aunt of Adanna. Oğuzhan, thanks for coming to the office late enough so I was never the last one arriving. You were not only a great office mate but also a great neighbor. Please move to Rotterdam and be my neighbor again. I also want to thank Majid for the valuable time we spent together. And the girl power of the office, Azqa, Huimin, Marina, and Miray, I wish to spend more time with you. It was difficult to be the only girl in the group for a long time.

Apart from the fun Ph.D. office, of course, I would like to acknowledge the members of the boring Ph.D. office. Harm, I wish you started your Ph.D. earlier, so we could annoy each other more. I know you always want to be in the fun Ph.D. room, I hope one day you will get there. Thanks a lot for all the help and all the support. I also would like to thank non-boring members of the boring Ph.D. office Vincent, Mark, Kris, Zhijie, and Qin.

During my Ph.D., I had the chance of supervising three master students. Leon, Hari, and Srinath, thanks for your hard work. Leon, I am really grateful to you for contributing to my Ph.D. thesis with your research. Hari, I am sorry that I missed your defense due to a very important event in Belfast.It is your turn now, and you have a valid excuse to miss my defense thanks to a pandemic that put all the world into a lockdown.

I also want to thank all members of Cyber Security Group: Christian, Sicco, Stjepan, Phil, Jan, Evgenie, Chris, and Laurens. And Sandra, the queen of the group, thanks for everything. It was really fun in the cybersecurity group with all the coffee breaks, lunches, and cake times. Having master students around made it even more fun. Thank you the ancients Bjorn, Christian, Dirk, Ginger, Hari, Lars, Leon, Mathijs, Mourad, Pieter, Prahesa, Rasmus, Rogier, Sjors, Victor and the newbies Daan, Wilko, Tim, Hugo, Maurits, and Jehan. Thank you, Rico, for the recipe of your coconut cake.

I want to particularly acknowledge the members of my dimple team, Lars and Victor, for all the joy, for visits to Five Guys, for the organization of the DimpleCon, and preparing my keynote speech on Schrodinger's dimple. Baran, it was always a pleasure to have endless discussions on the existential problems of life with you. The members of the multimedia group including Babak, Manel, Xiuxiu, and Jaehun, thanks for the coffee corner gatherings. The members of the computer graphics group, thanks for the apple times that gave us a healthy break among all the cake.

When I wanted to run away from the pressure of Ph.D. life, I had friends out of TU Delft that helped me with relaxing. Ezgi, I am so glad that I took a Dutch course, and I met you. It is always a pleasure for me to have vegetarian gatherings with you and spending time with Pati. You will always be my one and only Turkish female friend in Delft. Fatemeh thanks a lot for caring for me and being there whenever I need some support. I always like exploring new places with you and hope to have more of it. Nneka thanks for being an amazing host every Christmas but also for being my ultimate Pathe Unlimited partner.

Back in Turkey, I left great friends who were available online whenever I need. Nazlı and Güher, there are no words to describe my appreciation for our friendship. I am also grateful for all the trips we had during these years and I hope to have many more. Tuğçe and Ahmet, thanks for moving to the Netherlands and bringing the feelings of our bachelor years back. And thanks Melike, Ömer, Sevde, Alperen, Ahmet Can, Merve, Kardelen, Gökcen, Pınar, and Seray.

I owe big thanks to my parents Fatma and Mehmet, for all the support you provided throughout my education life. I know you got tired of me being a student for a long time but hopefully, we came to the end of it. I was lucky to have my brother Salih and my sister-in-law Zeynep close by during my Ph.D. studies. Thanks for opening your house to me whenever I need the feeling of family. Esin, my beloved niece, thank you for giving me the joy of being your aunt. And my sister Merve, thank you for all the long phone calls, laughter, stupid stories, and helping me with your artistic skills.

CURRICULUM VITÆ

Gamze TİLLEM

Gamze Tillem was born in Denizli, Turkey on February 6, 1990. She graduated from Denizli Erbakır Science High School in 2008. She obtained her bachelor's and master's degree in Computer Science and Engineering from Sabancı University in Istanbul in 2013 and 2015, respectively. She joined the Cyber Security Group of Delft University of Technology as a Ph.D. student in November 2015.

In 2008, she was ranked as 777^{th} in the national university entrance exam among one and a half million participants and was rewarded with an Honor Scholarship from Sabancı University. In her freshman year, she was nominated as one of the Massachusetts Institute of Technology – Sabancı University Freshman Scholars due to her success in Freshman courses. Her master's education was funded by TÜBİTAK (The Scientific and Technological Research Council of Turkey) BİDEB Graduate scholarship which she received as the result of success in her undergraduate GPA and ALES (the National Graduate Exam) score.

During her Ph.D., she was involved in Big Software on the Run project under the supervision of prof. dr. ir. R.L. Lagendijk and dr. Zekeriya Erkin. During her time in TU Delft, she supervised three master students and contributed to Privacy Enhancing Technologies and Security and Cryptography courses as a teaching assistant. She also visited EURECOM in France as a guest researcher and took part in PAPAYA project under the supervision of dr. Melek Önen.

As of December 2019, Gamze joined the Blockchain & DLT team at ING Bank as Dev-Engineer and Cryptographer.