Delft University of Technology

Efficient cryptographic building blocks for processing private measurements in e-healthcare

Nateghizad, Majid

DOI
10.4233/uuid:17c5457a-5fc7-420b-92a7-ad121d4b9fa9

Publication date
2019

Document Version
Final published version

Citation (APA)
Nateghizad, M. (2019). Efficient cryptographic building blocks for processing private measurements in e-healthcare. https://doi.org/10.4233/uuid:17c5457a-5fc7-420b-92a7-ad121d4b9fa9

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Efficient Cryptographic Building Blocks For Processing Private Measurements in e-Healthcare

Efficient Cryptographic Building Blocks For Processing Private Measurements in e-Healthcare

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof.dr.ir. T.H.J.J. van der Hagen,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op donderdag 28 november 2019 om 12:30 uur

door

Majid Nateghizad

Master of Computer Science in Information Security,
Universiti Teknologi Malaysia, Malaysia,
Geboren te Ghom, Islamic Republic of Iran

Dit proefschrift is goedgekeurd door de promotor

Prof. dr. ir. R.L. Lagendijk

Samenstelling promotiecommissie bestaat uit:

Rector Magnificus,	voorzitter
Prof. dr. ir. R.L. Lagendijk,	Technische Universiteit Delft, promotor
Onafhankelijke leden:	
Prof. Dipl.-Ing.dr. S. Katzenbeisser	University of Passau
Prof. dr. M. Petkovic	Eindhoven University of Technology
Prof. dr. ir. B.P.F. Lelieveldt	Technische Universiteit Delft
Prof. dr. ir. K.I. Aardal	Technische Universiteit Delft
Dr. ir. T. Veugen	TNO
Overig lid:	
Dr. Z. Erkin	Technische Universiteit Delft

Keywords: e-healthcare, privacy, multi-party protocol, building block, efficiency

Copyright © 2019 by M. Nateghizad

ISBN 978-94-6366-224-6

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

Neither any knowledge is like to seeking health
nor any health is like to health of heart.

Imam Baqir (p.b.u.h.)

Thanks to Allah Almighty who always put me in the right direction. Thanks to Imam Hussein (p.b.u.h.) who is giving me the opportunity to participate in ARBAEEN WALK with other millions of people each year. I am indebted to Imam Hussein (p.b.u.h.) for all my research achievements.
Thanks to my mother and father for supporting me these years. Thanks to my wife for being patient since I was working 7 days per week in the last three years. Thanks to my brothers Haji Rajabi, Sayyed Yaser, Alireza, Sayyed Mahmud Abbas, Amir and the others whom I cannot remember their names at this moment.
I have to acknowledge Dr. Erkin and Prof. Lagendijk for giving me the chance to study at TU Delft and helping me to accomplish my Ph.D. Coming to the Netherlands was a new beginning for me in terms of personal and professional life. I have experienced the best part of my life after starting my Ph.D. Thanks to Gamze, Chibuike, and Oguzhan for being inspiring and helping me to develop new building blocks.
I also have to acknowledge Prof. Petkovic for inspiring me to design the scenarios for this thesis, and Dr. Veugen for helping me to develop new building blocks and his comments on this thesis.

Contents

Summary xiii

Samenvatting xv

I Preface 1

1 Introduction 3
1.1 E-Healthcare . . . 3
1.2 Stakeholders . . . 4
1.3 Requirements of E-Healthcare . . . 5
1.4 Privacy Concerns in E-Healthcare . . . 6
1.4.1 Organizational Threats . . . 7
1.4.2 Systematic Threats . . . 8
1.5 Problem Statement . . . 8
1.6 Contributions . . . 9
1.7 Roadmap of This Thesis . . . 9
References . . . 11

2 Preliminaries 15
2.1 Multi-Party Protocols . . . 16
2.2 Security Settings . . . 16
2.2.1 Semi-Honest Model . . . 16
2.2.2 Malicious Model . . . 16
2.2.3 Covert Model . . . 16
2.3 Homomorphic Encryption . . . 17
2.3.1 Paillier Cryptosystem . . . 17
2.3.2 DGK Cryptosystem . . . 18
2.3.3 Fan-Vercanteren Cryptosystem . . . 18
2.4 Cryptographic Building-Blocks . . . 19
2.4.1 Secure Comparison Protocol . . . 19
2.4.2 Secure Equality Testing Protocol . . . 20
2.4.3 Data Packing . . . 21
References . . . 21

3 The Main Challenges in Developing Efficient and Secure e-Healthcare Systems 25
3.1 E-Healthcare Scenarios . . . 26
3.2 Centralized Single-Key Based E-Healthcare System (Scenario 1) . . . 27
3.2.1 Stake Holders . . . 27
3.2.2 Security and System Settings . . . 28
3.2.3 Challenges . . . 29

3.3 Centralized Multiple-Key Based E-Healthcare System (Scenario 2) 30
3.3.1 Stake Holders 30
3.3.2 Security and System Settings 31
3.3.3 Challenges 31
3.4 Decentralized Multiple-Key Based E-Healthcare System (Scenario 3) 32
3.4.1 Stake Holders 32
3.4.2 Security and System Settings 33
3.4.3 Challenges 33

4 Existing Applications of Cryptographic building blocks 35
4.1 Introduction 36
4.2 Existing Applications of Cryptographic Building Blocks 36
4.2.1 Medical Recommendation and Prediction Systems 36
4.2.2 Pattern Recognition 38
4.2.3 Classification and Clustering 39
4.2.4 Other Applications 40
4.3 Conclusion 43
References 43

II Secure Equality Testing Protocols 47

1 Efficient and Secure Equality Tests 49
1.1 Introduction 50
1.2 Preliminaries 50
1.2.1 Homomorphic Encryption 51
1.2.2 Security Assumptions and the Setting 51
1.3 Related Work 51
1.3.1 EQT Based on the Hamming Distance (LT13) 52
1.3.2 EQT Based on the Bit-Decomposition (ST06) 52
1.4 Improved Secure Equality Tests 52
1.4.1 Improved EQT Based on the Hamming Distance (NEL-I) 53
1.4.2 Improved EQT Based on the Bit-Decomposition (NEL-II) 53
1.5 Security Analysis 53
1.6 Performance Analysis 55
1.6.1 Computational Complexity 55
1.6.2 Experimental Results 57
1.6.3 Computation of α_i in NEL-I 57
References 58

2 Privacy-Preserving Equality Testing Protocols 61
2.1 Introduction 62
2.2 Preliminaries 63
2.2.1 Security Setting 63
2.2.2 Homomorphic Encryption 64

2.3 Our Protocols . . . 65
2.3.1 Equality Testing Protocol (EQT)-1 . . . 65
2.3.2 Equality Testing Protocol (EQT)-2 . . . 67
2.3.3 Equality Testing Protocol (EQT)-3 . . . 68
2.4 Security Analysis . . . 69
2.4.1 Security of EQT-1 . . . 70
2.4.2 Security of EQT-2 . . . 72
2.4.3 Security of EQT-3 . . . 74
2.5 Performance Analysis . . . 75
2.5.1 Complexity Analysis . . . 75
2.5.2 Experimental Results . . . 76
2.5.3 Applying Data Packing . . . 77
2.6 Conclusions . . . 79
References . . . 79

3 A Communication-wise Efficient Equality Testing Protocol 83
3.1 Introduction . . . 84
3.2 Preliminaries . . . 85
3.2.1 Notation . . . 85
3.2.2 System Setting . . . 85
3.2.3 Paillier Encryption Scheme . . . 86
3.2.4 Oblivious Transfer . . . 86
3.2.5 Oblivious Transfer Extension . . . 86
3.2.6 Ciampi and Orlandi's Private Set Membership Protocol . . . 87
3.3 Definition of Our Protocol . . . 88
3.4 Security Analysis . . . 89
3.5 Performance Analysis . . . 92
3.5.1 Complexity Analysis . . . 92
3.5.2 Experimental Verification . . . 95
3.6 Conclusion . . . 97
References . . . 98

III Comparison Protocol 101

1 An Efficient Comparison Protocol 103
1.1 Introduction . . . 104
1.2 Preliminaries . . . 105
1.2.1 Application Setting . . . 106
1.2.2 Security Model . . . 106
1.2.3 Homomorphic Encryption . . . 106
1.2.4 Paillier Cryptosystem . . . 106
1.2.5 DGK Cryptosystem . . . 107

1.3 Secure Comparison Protocol with Secret Inputs . . . 107
1.3.1 Computing $[z \bmod 2^{\ell}]$ . . . 108
1.3.2 Computing $[\lambda]$ . . . 108
1.3.3 Proposed Comparison Protocol . . . 109
1.3.4 Data Packing . . . 111
1.4 Performance Analysis . . . 111
1.5 Conclusion . . . 115
References . . . 115

IV Secure Searching and Retrieval **119**

1 Secure Index-Based Search Protocols **121**
1.1 Introduction . . . 122
1.2 Related Work . . . 123
1.3 Secure Searching Protocols . . . 125
1.3.1 IBSvI . . . 126
1.3.2 IBSvII . . . 130
1.4 Security Analyses . . . 131
1.4.1 Security of IBSvI . . . 131
1.4.2 Security of IBSvII . . . 134
1.5 Performance Analyses . . . 134
1.5.1 Complexity Analysis . . . 134
1.5.2 Experimental Results . . . 135
1.6 Conclusion . . . 137
References . . . 137

2 Knapsack Based Data Packing **141**
2.1 Introduction . . . 142
2.2 Preliminaries . . . 144
2.2.1 Subset Sum Problem and its Variations . . . 144
2.2.2 Additive Homomorphic Encryption . . . 144
2.3 Data Packing Based on Knapsack Problems . . . 145
2.3.1 Data packing using SISS . . . 145
2.3.2 Data Packing Using MTK Problem . . . 146
2.3.3 Data Packing Using CK Problem . . . 146
2.3.4 Modifying Packages . . . 147
2.3.5 Linear Operations Over Packages . . . 148
2.4 Performance Analysis . . . 148
2.4.1 Complexity of Initialization . . . 148
2.4.2 Complexity of Packing Encrypted Data . . . 148
2.4.3 Performance in Terms of ρ . . . 149
2.5 Conclusion . . . 150
References . . . 151

V Multiple Key Setting **153**

1 An Homomorphic Proxy Re-Encryption **155**
1.1 INTRODUCTION . . . 156
1.2 Related Works . . . 157
1.3 Preliminaries . . . 159
1.3.1 One-Direction Proxy Re-Encryption . . . 159
1.3.2 Correctness of Proxy Re-Encryption . . . 159
1.3.3 Public-Key Cryptosystem with a Double Trapdoor Decryption . . . 159
1.4 Homomorphic One-Direction Proxy Re-Encryption Scheme (HOPE) . . . 160
1.4.1 Correctness . . . 161
1.4.2 Homomorphism . . . 162
1.4.3 Data Packing in HOPE . . . 163
1.5 Security . . . 164
1.5.1 Computational Diffie-Hellman Problem (CDH) . . . 164
1.5.2 Lift Diffie-Hellman Problem (LDH) . . . 164
1.6 Performance Analysis . . . 165
1.7 Conclusion . . . 167
References . . . 167

VI Outlook **171**

1 Discussion and Future Work **173**
1.1 Discussion . . . 174
1.1.1 Core Building Blocks . . . 174
1.1.2 Index Based Data Filtering . . . 176
1.1.3 Data Packing . . . 177
1.1.4 Homomorphic Proxy Re-Encryption Scheme . . . 177
1.2 Future Work . . . 178
1.2.1 Scenario-1 . . . 178
1.2.2 Scenario-2 . . . 179
1.3 Conclusion . . . 179
References . . . 180

Summary

In order to achieve practical e-healthcare systems, five requirements should be addressed, namely 1) availability, 2) integrity, 3) accuracy, 4) confidentiality, and 5) efficiency. Using remote computer storage and processing services satisfies availability, integrity, and efficiency. However, it introduces privacy concerns regarding the leakage of private medical data to unauthorized parties, which violates GDPR. Data encryption is one of the widely used techniques to address those privacy concerns in e-healthcare systems. Although data encryption provides data confidentiality, while the accuracy and integrity of the data are preserved, it introduces computation and communication overheads that downgrade the efficiency of the e-healthcare systems.

To precisely find the bottlenecks in achieving privacy-preserving e-healthcare systems, we design three real-life e-healthcare scenarios. The scenarios are different in terms of the number of parties used in the system, the way that data are stored (centralized or distributed), and encryption key setting (single-key or multiple-key). Then, we identify the challenges and required cryptographic protocols for each scenario. Afterward, we investigate the performance of several applications that are using the same identified cryptographic protocols. We show that the existing cryptographic protocols, which are required for our scenarios, are dominating the computation and communication costs of the applications.

To address the challenges in the single-key setting, we improve the existing core building blocks, comparison, and equality testing, and develop new protocols to mitigate the overall costs of e-healthcare systems. We show that data filtering and retrieval protocols are still highly resource demanding, even though efficient building blocks are used. Thus, we develop a new secure indexing protocol that reduces the data filtering cost significantly. Moreover, we develop a novel data packing technique to achieve an efficient data retrieval protocol by using our indexing protocol. For the multiple-key setting, we introduce a homomorphic proxy re-encryption scheme. Our encryption scheme has several properties such as an unlimited number of re-encryption, supporting homomorphism after each re-encryption, one-direction re-encryption, and non-interactive re-encryption key generation. Afterward, we use our encryption scheme for data filtering in the multiple-key setting and evaluate its performance.

The results of the performance analysis of our protocols show that improving core building blocks can significantly decrease both computation and communication costs of the cryptographic applications. Moreover, we show that developing techniques such as data packing and indexing can limit the number of homomorphic operations considerably, and consequently, mitigate the overall computation and communication costs of the cryptographic applications.

Samenvatting

Om praktische e-gezondheidszorgsystemen te bereiken, moeten vijf vereisten worden aangepakt, namelijk 1) beschikbaarheid, 2) integriteit, 3) nauwkeurigheid, 4) vertrouwelijkheid en 5) efficiëntie. Het gebruik van externe computeropslag- en verwerkingsservices voldoet aan beschikbaarheid, integriteit en efficiëntie. Het introduceert echter privacy kwesties met betrekking tot het lekken van medische privégegevens aan onbevoegde partijen, hetgeen in strijd is met de AVG. Gegevenscodering is een van de meest gebruikte technieken om die privacy kwesties in e-gezondheidszorgsystemen aan te pakken. Hoewel gegevenscodering vertrouwelijkheid van gegevens biedt, terwijl de nauwkeurigheid en integriteit van de gegevens behouden blijven, introduceert het reken- en communicatieoverheadkosten die de efficiëntie van de e-gezondheidszorgsystemen verlagen.

Om de knelpunten bij het realiseren van privacy behoudende e-gezondheidszorg systemen precies te vinden, ontwerpen we drie realistische e-gezondheidszorgscenario's. Vervolgens identificeren we de uitdagingen en vereiste cryptografische protocollen voor elk scenario. Nadien onderzoeken we de prestaties van verschillende applicaties die dezelfde geïdentificeerde cryptografische protocollen gebruiken. We laten zien dat de bestaande cryptografische protocollen, die nodig zijn voor onze scenario's, de computer- en communicatiekosten van de applicaties domineren.

Om de uitdagingen in de single-key setting aan te gaan, verbeteren we de bestaande kern building blocks, vergelijking en gelijkheidstesten en ontwikkelen we nieuwe protocollen om de totale kosten van e-gezondheidszorgsystemen te verminderen. We laten zien dat datafilter- en ophaalprotocollen nog steeds zeer veeleisend zijn, ook al worden er vereiste building blocks gebruikt. Daarom ontwikkelen we een nieuw beveiligd indexeringsprotocol dat de kosten voor gegevensfiltering aanzienlijk verlaagt. Bovendien ontwikkelen we een nieuwe techniek voor het verpakken van gegevens om een efficiënt protocol voor het ophalen van gegevens te bereiken met behulp van ons indexeringsprotocol. Voor de instelling met meerdere sleutels introduceren we een homomorfisch proxy-hercoderings schema. Ons Encryptieschema heeft verschillende eigenschappen, zoals een onbeperkt aantal hercodering, ondersteuning van homomorfisme na elke hercodering, hercodering in één richting en het genereren van niet-interactieve hercoderingssleutels. Daarna gebruiken we ons coderingsschema voor het filteren van gegevens in de instelling met meerdere sleutels (multiple-key) en evalueren we de prestaties.

De resultaten van de prestatie-analyse van onze protocollen tonen aan dat het verbeteren van kern building blocks zowel de berekenings- als communicatiekosten van de cryptografische applicaties aanzienlijk kan verlagen. Bovendien laten we zien dat het ontwikkelen van technieken zoals het inpakken en indexeren van gegevens het aantal homomorfe bewerkingen aanzienlijk kan beperken en bijgevolg de totale berekenings- en communicatiekosten van de cryptografische applicaties kan verminderen.

I

Preface

I.1 | Introduction

Majid NATEGHIZAD

1.1. E-HEALTHCARE

Personal health information is becoming digitalized in order to facilitate sharing the data using ICT infrastructure. Fast growth of the amount of Electronic Health Records (EHR) is beneficial for improving public healthcare. Many types of research are using EHR to provide healthcare services [1–3]. As one example, Dual et al. [2] introduced a recommender system to improve clinical decision-making. As another example, Ryan Hoens et al. [3] developed a physician recommender system based on the given health conditions from patients. The amount of data collected from healthcare using devices like smart wearables and watches will reach to yottabyte (10^{24} bytes) scale in 2020 [4]. Moreover, this amount of data are collected from different sources, which makes healthcare data to be diverse in terms of velocity, volume, and variety [5]. Thus, healthcare systems are required to provide an environment to manage and process this amount of diverse data and make them real-time available for patients to be able to track and improve their well-being. Electronic Healthcare (e-healthcare) envisions efficient and effective healthcare services to improve public health through the incorporation of medical data and modern technological advances. It bridges the gaps like limited availability of patients and physicians that result in late or faulty decision-making in traditional healthcare systems. E-healthcare enables remote patient monitoring, assessment, and treatment, which lead to making a more informed decision at any time and place.

Since a lot of personal measurements (blood sugar, heart rate, etc.) with high accuracy are generated by patients using cutting-edge technology such as smart wearables [6], processing them is one of the important tasks of e-healthcare systems. That process includes making customized prediction [7], detection [8], and treatment [9] of diseases for patients. As an example, one of the services of e-healthcare systems is to generate customized health recommendation and statistics [1] to the patients, physicians, and other related research institutes. Recommender Systems (RecSys) play the main roles in e-healthcare systems since they help physicians and patients by generating customized and accurate recommendations to improve and speed up the decision-making process [10]. RecSys are capable of generating recommendations for different scenarios: from advising a list of highly-ranked physicians [3] based on a patient's symptoms to recommending customized diet and physical activity plans to a patient [11].

I.1

1.2. STAKEHOLDERS

There are several parties involved in different e-healthcare systems. In this thesis, three parties are appeared in our e-healthcare scenarios: 1) Patients, 2) smart device service providers, 3) medical institutes (hospitals).

- Patients: Patients are the owners of the measurements, which are stored in remote data storage units. The measurements can be blood pressure, heart rate, and blood sugar that are captured using smart devices and wearables. The collected data from each device are stored locally or kept in the vendors' databases. The data may contain additional information about the identity of the patients such as device type, location, date, time, etc.

- Smart device service providers (DSPs): DSPs are the vendors, who are offering various types of smart wearables to the patients and collect measurements from them in order to provide healthcare services. Maintaining a large amount of collected data from smart devices and processing them for generating statistics and recommendations demand both infrastructure and trained ICT staff and DSPs have limited computation and communication resources. Thus, using remote computation and storage services (cloud) are preferred to facilitate storing and processing measurements.

- Medical institutes: They are interested in analyzing patients' medical data stored in DSPs' databases or a cloud to monitor patients' well-being. Ideally, we would like the medical institutes to be able to use collected data for better prediction, prevention, and improvement of the well-being of the patients. However, similar to DSPs, they have limited computation and communication resources; thus, they use cloud resources.

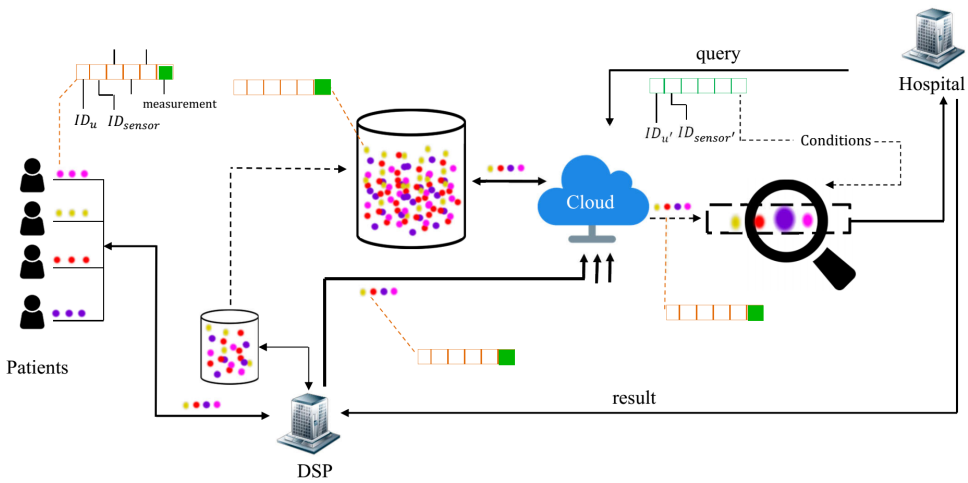

Figure I.1.1: Stackholders

The process of generating recommendation and statistics for the patients from a medical institute, as it is shown in Figure I.1.1, is as follows:

1. The smart medical devices capture the measurements and send them to their corresponding DSPs via smartphone.
2. DSPs store the given measurements from the patients in their local databases and send a copy of the measurements to the cloud.
3. Medical institute communicates with the cloud to obtain necessary information regarding one or a group of patients. Then, the obtained information is processed by the medical institute to generate the result such as recommendation and statistics.
4. Medical institute sends the result to the DSPs that are in contact with the target patients.
5. DSPs transfer the given result from medical institute to the target patients.

1.3. REQUIREMENTS OF E-HEALTHCARE

Although e-healthcare introduces tremendous benefits to patients and medical institutes, there are debates about how to deploy an e-healthcare system to mitigate the healthcare costs, how to share personal measurements, and how to secure them. In order to achieve an efficient e-healthcare system, several requirements need to be provided: 1) availability, 2) integrity, 3) accuracy, 4) confidentiality, and 5) efficiency.

- Availability: The main advantage of an e-healthcare system is the availability and accessibility of medical data. This property is necessary to track patients' well-being, detect any abnormal health condition, and take proper actions in real-time.
- Integrity: Ensuring integrity of medical data in its entire life-cycle is a must. That is because of having highly qualified and reliable patients' data for medical decision-making [12, 13]. Thus, data integrity should be preserved in different phases of e-healthcare systems such as storing, processing, retrieving, or securing patients' data.
- Accuracy: Using smart devices to measure vital signs like blood pressure and heart rate with high precision necessitates employing highly efficient and accurate techniques to analyze measurements. Using such techniques lead to obtaining reliable medical statistics and recommendations. Thus, none of the other requirements of e-healthcare systems should negatively affect the accuracy of the results.
- Confidentiality: The collected measurements are highly privacy-sensitive, which may consist of symptoms of diseases, personal data, etc. Moreover, the measurements that are being stored and processed in remote storage units are the potential targets for attackers. Thus, it is necessary to develop e-healthcare systems that protect patients' private data, while patients can receive healthcare services in real-time. Providing data confidentiality is of prime importance among other

requirements of e-healthcare systems. Lack of providing proper data confidentiality makes patients less willing to share their data with e-healthcare systems. In e-healthcare systems, it should be guaranteed that no private data is leaked to untrusted parties while patients' data are being stored, processed, or retrieved.

- Efficiency: Considering all the above requirements, achieving efficient e-healthcare systems that operate over large scale databases in real-time is vital. The efficiency means the amount of required resources such as computation and communication to run an e-healthcare system.

1.4. PRIVACY CONCERNS IN E-HEALTHCARE

In e-healthcare systems, patients need to share fine-grained measurements collected from smart wearables to e-healthcare systems. Then, they can receive correct diagnosis and treatment, and minimize adverse drug interactions. Other types of information also can be found from patients data such as identification, history of medical diagnosis and treatments, medical images, genetic information, employment history, and income. Figure I.1.2 shows how health information is shared among parties for different purposes. The purposes can be improving healthcare decision-making quality, updating public policy, adjusting insurance and medical costs, and improving care services by health information organizations. Companies such as private insurance are also interested to access patients' data to justify their payments for the services.

Considering the medical data flow, achieving an e-healthcare system in practice that meets all the requirements is challenging, since satisfying one of the requirements may negatively affect the others.

- To satisfy the availability, as one of the requirements of e-healthcare systems, e-healthcare systems benefit from powerful ICT infrastructure by third-parties. They facilitate handling a large amount of medical data with a high level of availability and efficiency.

- The medical data should not be altered for any reason since it affects the accuracy of the results of e-healthcare systems. Generating unreliable statistics and recommendations to physicians and patients might put the patients' well-being in danger. Moreover, medical institutes are interested in generating customized recommendations for the patients; therefore, storing personal identity information alongside the medical measurements in the remote storage is necessary.

- Storing the medical data in remote storage service providers may violate the privacy of patients' medical data [14]. Patients may deny providing information such as psychiatric behavior or types of cancer to the system, since disclosure of their private data may result in social stigma and discrimination [15, 16]. Thus, securing the data before outsourcing is a must. Moreover, the securing technique should not affect the integrity and accuracy of medical data. The generated statistics and recommendations from secured data should be similar to the case when they are generated from data in clear form.

- The secured data in remote storage service providers should be in such form that is possible to process them and generate the desired results in real-time. Although it may not be feasible to achieve the same efficiency as processing clear data, the overhead computation and communication costs should be minimized.

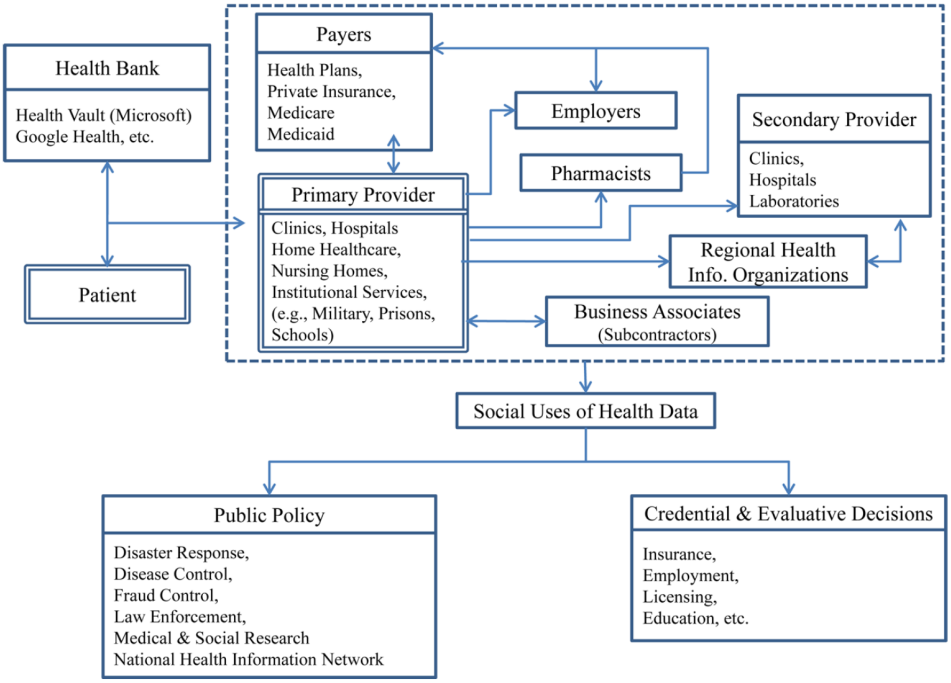

Figure I.1.2: Information flow in the health care system [17]

To show that why protecting data confidentiality in e-healthcare systems is a must, we list several types of threats [18, 19] as follows:

1.4.1. ORGANIZATIONAL THREATS

Organizational threats put the patients' data under risk of data disclosure through unauthorized or inappropriate access to the database. Inappropriate access control system and vulnerabilities against external attacks are examples of security gaps that enable organizational threats. The level of damage to organizational threats might vary depending on the motivation, available financial resources, and accessibility of attackers. The research in [19] shows that the damage to organizational threats can be categorized under five levels:

- Accidental disclosure: Unintended healthcare data breach by healthcare service providers and personnel. The level of data disclosure at this level may not be very severe. Sending part of patients' data unintentionally to other people can be an example of this threat level.

- Insider curiosity: An employee uses his legitimate data-access to seek for private data of patients like celebrities to share it with media. As the name stands for, the target of the threat is generally one individual or a group of people of interest.
- Data breach insiders: This threat is similar to insider curiosity, where a larger group of people may be targeted and more frequently. The insider may leak the privacy-sensitive patients' data to an outsider for financial profit or to foreign intelligence agencies.
- Data breach through physical intrusion: In this level of threat, an outsider or unauthorized personnel get access to the physical facilities. The amount of data leakage depends on the protection layers applied to the system and the private data.
- Data breach outsiders: Getting into the organizations' networks by hacking or using insiders to intercept private communication is also another way to access patients' data.

The consequences of the above threats are not only limited to disclosure of patients' data, but also availability, integrity, and performance of a system can be the targets.

1.4.2. Systematic Threats

Systematic threats are mostly from foreign governments and well-heeled organizations. Communities of systematic threats can be insiders and outsiders. It can be observed that the majority of privacy violation of patients' data happened by insiders who are legally authorized to access patients' data. To clarify the motive, insurance companies are a well-known example that spend a lot of money to obtain patients' medical data. This information helps insurance companies to save money by knowing customers with high-risk diseases.

1.5. Problem Statement

In this thesis, we are envisioning privacy-preserving e-healthcare systems in which medical institutes can use the measurements that are collected from smart devices for improving the well-being of patients. Medical institutes can generate customized statistics and medical recommendation for patients based on the given measurements. However, there are privacy concerns in e-healthcare systems regarding the confidentiality of patients' privacy-sensitive data when it is stored and processed in third parties. To address those privacy concerns, various techniques based on data access control, data anonymity and generalization, or data encryption are introduced.

In this thesis, we are focusing on cryptographic solutions to fill that privacy gap of e-healthcare systems in multi-party settings. Using cryptographic solutions in e-healthcare has three advantages as follows:

- Data encryption provides data confidentiality by securing the data itself using technological methods.
- Data encryption keeps the data intact and in its full form, which is a must in the medical domain to generate accurate recommendation and statistics. It is also

important to have patients' personal information to be able to contact them in case of emergency.

- Homomorphic encryption schemes enable computation of data while they are in the encrypted form. This property helps to improve the performance of e-healthcare systems when a third-party is processing encrypted data. Moreover, by using homomorphic data encryption schemes, it is possible to generate statistics and recommendations with the accuracy similar to the case where the data are not encrypted.

Using homomorphic data encryption in e-healthcare satisfies all the requirements of e-healthcare, but the efficiency. The size of the data can become hundreds of times larger after applying data encryption. Data expansion introduces a significant amount of computation and communication costs for storing and analyzing privacy-sensitive data. These costs can vary considerably based on the system and security configurations and they can become a serious challenge against using data encryption in e-healthcare systems with large databases. The key questions posed in this thesis address the challenges of processing large set of encrypted measurements. The research questions are:

- How can changing system and security configuration in e-healthcare affect security and performance?
- How should the challenge of resource-demanding cryptographic applications be addressed effectively?
- Is it possible to achieve practical privacy-preserving e-healthcare systems using homomorphic encryption?

1.6. CONTRIBUTIONS

In this thesis, we design three secure real-life e-healthcare scenarios that are different in system and security settings. These scenarios help to investigate and identify the bottlenecks regarding the computation and communication costs when homomorphic data encryption is used to protect and process the medical data in multi-party settings. Afterward, we show that improving the core building blocks improve the total performance of cryptographic solutions considerably. Then, we improve the core building blocks of the existing cryptographic protocols used in e-healthcare. Moreover, we introduce novel cryptographic protocols and an encryption scheme that can be used to boost the efficiency and functionality of any application that is relying on cryptographic solutions.

1.7. ROADMAP OF THIS THESIS

In the following, an overview of the structure of the thesis is given.

Chapter I.2 In this chapter, we describe three security settings, where the cryptographic protocols should be secured according to. Then, we explain the cryptographic primitives that are used in this thesis, which includes three homomorphic encryption schemes. Afterward, we explain the core building blocks and data packing technique

that are employed in our cryptographic protocols. The core building blocks include comparison and equality testing protocols.

Chapter I.3 In this chapter, we explain three different e-healthcare scenarios that are different in terms of system and security settings. Then, we show that what the challenges are to realize the scenarios. Based on the identified challenges, we detect the most effective building blocks and cryptographic protocols that play essential roles in the overall performance of the scenarios.

Chapter I.4 This chapter includes a brief description of existing applications of core cryptographic building blocks, secure data packing, and secure searching. Then, we investigate their performance and show how much the core building blocks are contributing to the total performance of the existing applications.

Chapter II.1 In this chapter, we address one of the core building blocks, **secure equality testing**. First, we choose two state-of-the-art equality testing protocols. Second, we find the bottlenecks in the protocols and propose improvements to address them. Then, we apply the improvements and evaluate the performance of the new protocols. This chapter is an integral copy of "Efficient and Secure Equality Tests", by M.Nateghizad, Z.Erkin and R.L.Lagendijk in the proceedings of 8th *International Workshop on Information Forensics and Security*, 2016. Note that this paper does not provide a formal security proof for the developed protocols.

Chapter II.2 This chapter addresses the trade-off between computation and communication costs in the **secure quality testing** protocols. Because applications may have different available resources in terms of computation and communication, we introduce three different equality testing protocols. Each protocol has a different trade-off between computation and communication to meet different system requirements. This chapter is an integral copy of "Secure Equality Testing Protocols in the Two-Party Setting", by M.Nateghizad, T.Veugen, Z.Erkin and R.L.Lagendijk in the proceedings of 13th *International Conference on Availability, Reliability and Security*, 2018. This work was nominated for the best paper award.

Chapter II.3 This chapter is more focused on developing a **secure equality testing** protocol with highly efficient communication cost. By providing experimental results, we show that the communication cost of the protocol is significantly more efficient than the other introduced protocols with a competitive computation cost. This chapter is an integral copy of "SET-OT: A Secure Equality Testing Protocol Based on Oblivious Transfer", by F.Karakoç, M.Nateghizad and Z.Erkin to the 14th *International Conference on Availability, Reliability and Security*, 2019.

Chapter III.1 This chapter addresses the challenges of the existing **secure comparison** protocols, as one of the core building blocks. We address two different system settings that are 1) two parties with private inputs and 2) two parties with encrypted inputs. To achieve a privacy-preserving comparison protocol that is efficient in both system settings, we improve the core part of the existing state-of-the-art comparison protocol. Then, with the help of data packing, we show that our new protocol is considerably more efficient than existing works in both settings. This chapter is an integral copy

of "An efficient privacy-preserving comparison protocol in smart metering systems", by M.Nateghizad, Z.Erkin and R.L.Lagendijk in the *EURASIP Journal of Information Security*, 2016. Note that a formal security proof for the developed protocol in this paper is provided in Chapter II.2.

Chapter IV.1 This chapter addresses the challenge of filtering encrypted data. We develop a novel **index based data filtering and searching** using an additively homomorphic encryption scheme. Afterward, we show that non-interactive secure data filtering can be achieved by using somewhat data encryption. Then, we compare the performance of our protocols in terms of computation and communication costs with state-of-the-art protocols. This chapter is an integral copy of "Efficient Index-based Search Protocols for Encrypted Databases", by M.Nateghizad, Z.Erkin, and R.L.Lagendijk in the proceedings of 15th *International Joint Conference on e-Business and Telecommunications*, pages 436-447, 2018.

Chapter IV.2 Data packing is a useful tool to decrease both computation and communication costs of cryptographic applications. In this chapter, we developed novel **data packing** techniques based on trapdoor knapsack problem. Then, we apply our data packing on our index-based data filtering to achieve an efficient **secure data retrieval** protocol. This chapter is an integral copy of "A Novel Approach For Data Packing: Using Trapdoor Knapsack", by M.Nateghizad, Z.Erkin, and R.L.Lagendijk in the proceedings of 10th *International Workshop on Information Forensics and Security*, 2018.

Chapter V.1 In this chapter, we introduce an **homomorphic proxy re-encryption scheme**. We show that our scheme supports more features than the existing works. Then, through experimental results, we show that our protocol outperforms the state-of-the-art. This chapter is an integral copy of "HOPE: A Homomorphic One-Direction Proxy Re-Encryption Scheme", by M.Nateghizad, Z.Erkin, and R.L.Lagendijk to be sumbitted to 11th *International Workshop on Information Forensics and Security*, 2019.

This thesis consist of a series of integral copies of published and submitted papers by the author. Therefore, there are inconsistency in the used notation, overlaps in terms of motivation, and explanation on the building blocks. However, for improving the clearance and correctness of the thesis, the typos in some of the published papers are corrected in this thesis. Moreover, new references are provided for the statements that demand further explanation.

REFERENCES

[1] M. Wiesner and D. Pfeifer, *Health recommender systems: concepts, requirements, technical basics and challenges,* International journal of environmental research and public health **11**, 2580 (2014).

[2] L. Duan, W. N. Street, and E. Xu, *Healthcare information systems: data mining methods in the creation of a clinical recommender system,* Enterprise IS **5**, 169 (2011).

I.1

[3] T. R. Hoens, M. Blanton, A. Steele, and N. V. Chawla, *Reliable medical recommendation systems with patient privacy,* ACM TIST **4**, 67:1 (2013).

[4] K. Corbin, *How CIOs Can Prepare for Healthcare 'Data Tsunami'* (2014), available from https://www.cio.com/article/2860072/healthcare/how-cios-can-prepare-for-healthcare-data-tsunami.html.

[5] M. Khan, B. N. Silva, and K. Han, *Efficiently processing big data in real-time employing deep learning algorithms,* in *Deep Learning Innovations and Their Convergence With Big Data* (IGI Global, 2018) pp. 61–78.

[6] W. H. Organization *et al.*, *Medical devices and eHealth solutions: compendium of innovative health technologies for low-resource settings 2011-2012* (World Health Organization, 2013).

[7] Y. Cheng, F. Wang, P. Zhang, and J. Hu, *Risk prediction with electronic health records: A deep learning approach,* in *Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, Florida, USA, May 5-7, 2016* (2016) pp. 432–440.

[8] A. C. Cheng, *Real-time cardiovascular diseases detection on a smartphone,* Departments of Electrical and Computer Engineering, Bioengineering, Neurological Surgery and Computer Science (2011).

[9] T. D. Sequist, T. K. Gandhi, A. S. Karson, J. M. Fiskio, D. Bugbee, M. Sperling, E. F. Cook, E. J. Orav, D. G. Fairchild, and D. W. Bates, *Technology evaluation: A randomized trial of electronic clinical reminders to improve quality of care for diabetes and coronary artery disease,* JAMIA **12**, 431 (2005).

[10] T. B. Murdoch and A. S. Detsky, *The inevitable application of big data to health care,* Jama **309**, 1351 (2013).

[11] P.-Y. Benhamou, *Improving diabetes management with electronic health records and patients' health records,* Diabetes & metabolism **37**, S53 (2011).

[12] D. P. Lorence, A. Spink, and R. Jameson, *Information in medical decision making: How consistent is our management?* Medical Decision Making **22**, 514 (2002).

[13] D. Lorence, A. Spink, and R. Jameson, *Assessing managed care market variation in reports of coding accuracy.* Managed care quarterly **10**, 15 (2002).

[14] R. Mercuri, *The hipaa-potamus in health care data security,* Commun. ACM **47**, 25 (2004).

[15] P. Applebaum, *Privacy in psychiatric treatment: threats and response,* American Journal of Psychiatry **159**, 1809 (2002).

[16] P. Sankar, S. Mora, J. F. Merz, and N. L. Jones, *Patient perspectives of medical confidentiality: a review of the literature,* Journal of general internal medicine **18**, 659 (2003).

[17] A. Appari and M. E. Johnson, *Information security and privacy in healthcare: current state of research,* International Journal of Internet and Enterprise Management **6**, 279 (2010).

[18] T. C. Rindfleisch, *Privacy, information technology, and health care,* Commun. ACM **40**, 92 (1997).

[19] J. D. Halamka, P. Szolovits, D. M. Rind, and C. Safran, *Application of information technology: A WWW implementation of national recommendations for protecting electronic health information,* JAMIA **4**, 458 (1997).

I.2 | Preliminaries

In this chapter security settings, and cryptographic primitives and tools to build an e-healthcare are presented. First, we explain three different security notations semi-honest, covert, and malicious. Second, the cryptographic primitives that are used in this thesis are presented. Then, we explain cryptographic building-blocks, secure comparison protocol and equality test that repeatedly used in the privacy-preserving e-healthcare systems. Afterward, we describe data packing and its effect on improving the performance of cryptographic protocols.

2.1. MULTI-PARTY PROTOCOLS

In multi-party protocols, two or more parties computing a multi-variable function on their private inputs. However, the parties do not want to share their private with each others. For more information, we refer readers to [1].

2.2. SECURITY SETTINGS

2.2.1. SEMI-HONEST MODEL

In a multi-party protocol, the semi-honest security model is defined as each party follows the protocol instructions as it is agreed. However, parties are curious to learn more about private data than they are entitled to. This security setting has different names such as passive, honest-but-curious, semi-honest. In this security setting, parties store all the intermediate messages and computations in order to infer as much information as they can [2]. This setting can be realistic in many business models where third parties like companies providing cloud services care about their reputation. Comparing with the other two security settings, malicious and covert, cryptographic protocols under semi-honest security assumption are more efficient for large scale applications. Protecting confidential data in malicious and covert settings demand using additional security primitives in the protocols, which negatively affect their performance regarding computation and communication costs.

2.2.2. MALICIOUS MODEL

Malicious activity of a party in a multi-party protocols can be defined in three statements: 1) Parties can decide whether to join the computation. 2) parties can abort or suspend the protocol at any time and any step of the protocol instructions. This abortion may happen when the adversary receives the desired message from the other parties in the middle of the protocol. 3) It is not clear whether parties insert the correct local inputs into the protocol or generate fake inputs [2]. Protocols that are secure in malicious setting require more computation and communication resources compared to the same solutions in semi-honest model.

2.2.3. COVERT MODEL

In this setting, the adversary plays a role between semi-honest and malicious. He intends to actively cheat as long as he is not caught. This security setting can be more close to reality since many companies are willing to gain as much data as they can and cheat in following the protocol instructions. It can also be the case that adversary considers the risk of being caught, his benefit of cheating in the protocol, and then decide to cheat. Although covert security setting differs from the malicious setting, the protocols should be designed to be secure against any possible attack from other parties in the system.

In this thesis, we choose the semi-honest security setting, since it is publicly accepted in the research literature. Designing and developing secure systems that are resistant to malicious attacks demand using more complicated protocols, which significantly reduce the performance of systems [3]. Moreover, the assumption of malicious activities of companies and organization may not be very close to reality. The reason is that companies and organization care about their reputation in the business world. Thus any

malicious cyber activities may result in losing their markets. Unlike the malicious setting, covert security setting can be a more realistic assumption in theory than other two security settings based on its definition. However, in this thesis, our focus is on the semi-honest setting because of actual needs in real cases of interest.

2.3. HOMOMORPHIC ENCRYPTION

In this thesis, we rely on homomorphic cryptosystems such as Paillier [4] and DGK (Damgård, Geislet and Krøigaard) [5], and Fan-Vercanteren (FV) [6]. Additively homomorphic encryption schemes preserve a certain structure that can be exploited to process ciphertexts without decryption. Given $\mathscr{E}_{pk}(m_1)$ and $\mathscr{E}_{pk}(m_2)$, where pk is the public key, a new ciphertext whose decryption yields the sum of the plaintext messages m_1 and m_2 can be obtained by performing a certain operation over the ciphertexts:

$$\mathscr{D}_{sk}(\mathscr{E}_{pk}(m_1)) \otimes (\mathscr{E}_{pk}(m_2)) \mod n = m_1 + m_2 \,. \tag{2.1}$$

Consequently, exponentiation of any ciphertext with a public value yields the encrypted product of the original plaintext and the exponent:

$$\mathscr{D}_{sk}(\mathscr{E}_{pk}(m)^e) \mod n = e \cdot m \,. \tag{2.2}$$

Unlike the additively homomorphic encryption schemes, somewhat fully homomorphic schemes enable unlimited time performing of one linear operation whether addition and multiplication and a limited number of the other linear operation. For example, the homomorphic encryption scheme in [6] allows an unlimited number of additions and a few numbers of multiplications. There are also other homomorphic encryptions that allow both addition and multiplication to be performed an unlimited number of times at the cost significantly higher computational and communication costs.

2.3.1. PAILLIER CRYPTOSYSTEM

The Paillier encryption function for a given message $m \in \mathbb{Z}_\eta$ is defined as follows:

$$c = \mathscr{E}_{pk}(m, \tau) = g^m \cdot \tau^\eta \mod \eta^2 \,, \tag{2.3}$$

where η is the product of two distinct large prime numbers p and q, ciphertext $c \in \mathbb{Z}^*_{\eta^2}$, $\tau \in_R \mathbb{Z}^*_\eta$ and $g \in \mathbb{Z}^*_{\eta^2}$ is a generator of order η. The decryption function is,

$$\frac{L_\eta(c^{\lambda_\eta} \mod \eta^2)}{L_\eta(g^{\lambda_\eta} \mod \eta^2)} \mod \eta = m \,, \tag{2.4}$$

where λ_η is the Carmichael value that is the smallest positive integer such that $\{\forall a \in \mathbb{Z}^*_\eta : a^{\lambda_\eta} \equiv 1 \,(\mathrm{mod}\ \eta)\}$ and $L_\eta(x) = \frac{x-1}{\eta}$. The public key is (g, η) and the private key is λ_η.

The homomorphic property can be shown as below:

$$\begin{aligned} &\mathscr{D}_{sk}((\mathscr{E}_{pk}(m_1)) \times (\mathscr{E}_{pk}(m_2)) = \mathscr{D}_{sk}(g^{m_1} \cdot \tau_1^\eta \times g^{m_2} \cdot \tau_2^\eta) \\ &= \mathscr{D}_{sk}(g^{m1+m2} \cdot (\tau_1 \cdot \tau_2)^\eta) = \mathscr{D}_{sk}(\mathscr{E}_{pk}(m_1 + m_2)) \mod \eta = m_1 + m_2 \,. \end{aligned} \tag{2.5}$$

2.3.2. DGK CRYPTOSYSTEM

We also use the DGK cryptosystem [5, 7], which is used in constructing cryptographic protocols because of its efficiency due to its small message space.

For generating the public and the private keys, there are three parameters: k, t, and ℓ, where $\ell < t < k$. The process of key generation is as follows:

1. Choose two distinct t-bit prime numbers v_p, v_q.
2. Construct two distinct prime numbers p and q, where $v_p|(p-1)$ and $v_q|(q-1)$ such that $n = pq$ is a k-bit RSA modulus.
3. Choose u as the smallest possible prime number such as 8-bit value as suggested in [8].
4. Choose a random r that is longer that $2t$-bit.
5. Choose $(g, h) \in \mathbb{Z}_{n^*}$ such that g has order uv_pv_q and h to be of order v_pv_q.

The public and the private keys are $pk = (n, g, h, u)$ and $sk = (p, q, v_p, v_q)$, respectively.

The encryption of a plaintext $m \in \mathbb{Z}_u$ is given as follows:

$$c = \mathscr{E}_{pk}(m, r) = g^m \cdot h^r \mod n. \tag{2.6}$$

To decrypt the ciphertext one can build a look-up table for all $m \in \mathbb{Z}_u$ values and obtain m from $c^{v_p} \mod p = (g^{v_p})^m \mod p$. DGK scheme can efficiently check whether a ciphertext is an encryption of zero or not. To achieve this, we check whether $c^{v_pv_q} \mod n = 1$ or more efficiently we only need to prove that $c^{v_pv_q} \mod p = 1$ or $c^{v_pv_q} \mod q = 1$, since $u < p$ [5].

2.3.3. FAN-VERCANTEREN CRYPTOSYSTEM

In general FV scheme has seven algorithms that are briefly explained as follows:

1. *SecKeyGen*(λ): Let λ be the security parameter, *SecKeyGen* samples a uniformly distributed random number s from $\mathbb{R}_2$ and outputs $sk = s$, $s \xleftarrow{\$} \mathbb{R}_2$, where $\mathbb{R}_a$ is a ring $\mathbb{Z}_a[x]/(x^n+1)$ and $\xleftarrow{\$}$ refer to choosing a random number.
2. *PubKeyGen*(sk): Chooses a uniformly distributed random number a from $\mathbb{R}_q$, $a \xleftarrow{\$} \mathbb{R}_q$, and $r \leftarrow \mathscr{X}$, where $\mathscr{X}$ is a truncated discrete Gaussian distribution. It outputs $pk = ([-(as+e)]_q, a)$. $[x]_q$ shows a mapping that reduces $x \in \mathbb{Z}_q$ such that $[x]_q = a \mod q$.
3. *EvalKeyGen*(sk, w): Let w be a base into which ciphertexts are decomposed in relinearization, it chooses a random number $a_i \xleftarrow{\$} \mathbb{R}_q$ and $e_i \xleftarrow{\$} \mathscr{X}$ for $i \in \{0, \cdots, \ell\}$, where $\ell = \lfloor \log_w q \rfloor$. Then, it outputs $evk(p_0, p_1) = ([-(a_i s + e_i) + w^i s^2]_q, a_i)$.
4. *Encrypt*(pk, m): It takes $m \in \mathbb{R}_t$ (plaintext space), $u \xleftarrow{\$} \mathbb{R}_2$, and $e_1, e_2 \leftarrow \mathscr{X}$. Then, it computes the ciphertext ct such that $ct(ct[0], ct[1]) = ([\Delta m + p_0 u + e_1]_q, [p_1 u + e_2]_q)$, where Δ is $\lfloor q/t \rfloor$.

5. *Decrypt*(sk, ct): It computes $m = [\lfloor \frac{t}{q}[ct[0] + ct[1]s]_q \rceil]_t$

6. *Add*(ct_0, ct_1): It outputs $Encrypt(pk, m_0 + m_1) = (ct_0[0] + ct_1[0], ct_0[1] + ct_1[1])$.

7. *Multiply*(ct_0, ct_1): Multiplication of two ciphertexts is more complicated than the addition. First, it computes $c_0 = [\lfloor \frac{t}{q} ct_0[0] ct_1[0] \rceil]$, $c_1 = [\lfloor \frac{t}{q}(ct_0[0]ct_1[1] + ct_0[1]ct_1[0]) \rceil]$, and $c_2 = [\lfloor \frac{t}{q} ct_0[1] ct_1[1] \rceil]$. Then, it represents c_2 in base w as $c_2 = \sum_{i=0}^{\ell} c_2^{(i)} w^i$, where ℓ represents the number of elements in each component of each evaluation key. Finally, it computes $\acute{c_0} = c_0 + \sum_{i=0}^{\ell} evk[i][0] c_2^{(i)}$ and $\acute{c_1} = c_1 + \sum_{i=0}^{\ell} evk[i][1] c_2^{(i)}$, and outputs $(\acute{c_0}, \acute{c_1})$ as the result of the multiplication. For further explanation, we refer readers to [6].

In FV scheme, the plaintext and ciphertext space are $\mathbb{R}_t$, and $\mathbb{R}_q \times \mathbb{R}_q$, respectively. $\mathbb{R} = \mathbb{Z}[x]/(x^n + 1)$ is a polynomial of degree less than n with coefficients modulo t and (t, q) should be chosen such that $t \ll q$. The security of the FV scheme is based on the hardness of the Learning with Errors (LWE) problem.

Recall that message space and ciphertext space are constructed by using ring $\mathbb{R} = \mathbb{Z}[x]/(x^n + 1)$, where n is always a power of 2. By choosing t such that the polynomial modulus $(x^n + 1)$ factors into n linear factors modulo t, $2n|(t-1)$ [6]. This makes plaintext space $\mathbb{R}_t$ to break into the product as $\mathbb{R}_t \cong \mathbb{Z}_t^n$, which means it supports both multiplication and addition. This technique is called batching, which enables FV scheme to pack multiple messages into one plaintext such that addition and multiplication can be performed over its encryption. Batching can improve the performance of cryptographic applications significantly when performing homomorphic operations over ciphertexts in large datasets.

2.4. CRYPTOGRAPHIC BUILDING-BLOCKS

2.4.1. SECURE COMPARISON PROTOCOL

Yao [9] first introduced the problem of comparing two values without leaking information about the values. Secure comparison protocols are one of the most repeated core building-blocks in secure applications such as face recognition [10], finger-code authentication [11], and K-means clustering [12]. We consider two different settings for comparison protocol to be deployed, 1) secure comparison protocol with private inputs and 2) secure comparison protocol with encrypted inputs. In the first setting, it is assumed that there are two parties in the setting where each one is holding a private input. These private inputs must be only known to their owners and should not be revealed to the other parties during the process of comparison protocol. At the end of the protocol, both parties will only learn who has the bigger value and they do not obtain more information than the comparison result. In the second setting, there are two parties in the system, where one of the parties holds two encrypted values and another party has the decryption key [10]. In this setting, the goal is to compare two encrypted values such that the result is encrypted form and none of the parties learn any information about the encrypted values, any relation between two encrypted values, or the comparison result. This goal is more challenging to achieve than the goal in the first setting, because the inputs are in encrypted form and the result should be also obtained in encrypted form.

As an example, we show the construction of secure comparison protocol introduced in [10].

Let us assume that there are two parties in the system, party A and party B, where party A has two encrypted values, $[a]$ and $[b]$, under additively homomorphic encryption and party B hold the decryption key. The comparison protocol in [10] relies on the fact that $[2^{-\ell} \cdot (z - (z \bmod 2^{\ell}))]$, where $z = 2^{\ell} + a - b$ and ℓ is the bit length of the inputs, gives the encrypted solution for comparison protocol. However, the challenge is to compute $[z \bmod 2^{\ell}]$ which is explained in [10] as follows:

1. Party A chooses a random value r and computes $[d] = [z + r]$ and sends $[d]$ to B.

2. Party B decrypts the given ciphertext $[d]$ to obtain d and then computes $\hat{d} = d \bmod 2^{\ell}$ and sends the encryption of bits $[\hat{d}_i]$ to A.

3. Party A computes $\hat{r} = r \bmod 2^{\ell}$, and then $[c_i] = [\hat{d}_i - \hat{r}_i + s + 3\sum_{j=i+2}^{\ell-1} w_j]$, where $[w_j] = [\hat{d}_j \oplus \hat{r}_j]$, $i \in \{0, \cdots, \ell - 1\}$, and s is randomly choosen from $\{1, -1\}$. Adding the values s in computation of c_i prevents information leakage regarding the result of the comparison protocol.

4. Party A multiplicatively masks $[c_i]$ by choosing radom number r_i and performing $[e_i] = [c_i \cdot r_i]$. Then, A send $[e_i]$ to B.

5. Party B decrypts $[e_i]$ and then checks if any one them is zero. B creates an encrypted bit $\acute{\lambda}$ based on finding any zero and sends it to A.

6. Party A fixes $\acute{\lambda}$ based on s to obtain λ and then computes $[z \bmod 2^{\ell}] = [\hat{z} + \lambda 2^{\ell}]$, where $\hat{z} = \hat{d} - \hat{r}$.

2.4.2. SECURE EQUALITY TESTING PROTOCOL

Another core cryptographic protocol is secure equality testing. Similar to the secure comparison protocol, this protocol is also one of the tools that is used in many cryptographic applications with high number of repetition. From finding similar users in a system with millions of users to a particular one based on his or her taste for movies [13], secure pattern matching [14], and secure linear algebra [15] to encryption switching protocols [16] are some of the applications of secure equality testing protocol. Secure equality testing protocols can be used in two settings: 1) there are two inputs, where each holds a private and unencrypted values, and 2) one party holds two encrypted values and another party has the private key. The security requirements of secure equality testing in two settings are similar to the requirements of secure comparison protocol. For better understanding of how an equality testing protocol works, the construction of a secure equality testing protocol from [17] is presented as follows:

1. Party A computes $[z] \leftarrow [a - b]$, masks the result additively with a random number r, $[x] \leftarrow [z + r]$, and sends it to Party B.

2. Party B decrypts $[x]$, picks the first ℓ less significant bits, encrypts them separately, and sends them to party A.

3. Party A computes the Hamming distance d between x and r. d becomes zero if and only if $a = b$, since in this case $z + r = r$. Afterwards, party A masks $[d+1]$ multiplicatively with the inverse of a random number R and sends the masked ciphertext, $[y] \leftarrow [d+1]^{R^{-1}}$, to party B. Party A adds one to d to make sure $d+1 \in \mathbb{Z}_n^*$.

4. Party B decrypts y and computes the exponentiations y^i, $1 < i \leq \ell$. Then, party B encrypts y^i and sends them back to party A.

5. Party A unmasks the $[y^i]$ to obtain $[d^i]$ by computing $[d^i] \leftarrow [y^i]^{R^i}$ and computes ℓ-degree Lagrange polynomial $\vartheta(x)$, and from that $\vartheta(d)$ maps $d = 1$ to 1, and $d \in \{2, 3, \ldots, \ell\}$ to 0.

2.4.3. DATA PACKING

The main idea behind data packing [8, 18] is to efficiently use the message space of the encryption system, such as Paillier, in a protocol. Assume $[a]$ is the encryption of an ℓ-bit integer, and n is the message space of Paillier encryption system. Party A can pack $\rho = \lfloor \log n / \log \ell \rfloor$ different $[a]_i$ into one Paillier encryption as follows:

$$[\hat{a}] = \sum_{i=0}^{\rho-1} [a]_i^{(2^\ell)^i}. \tag{2.7}$$

Afterwards, party A sends $[\hat{a}]$ to party B, who computes decrypts and unpacks it. Employing data packing technique not only improves the efficiency of Paillier decryption, but also decreases total data transmission between two parties substantially.

There are two approaches to use data packing based on the system setting: 1) packing clear data and 2) packing encrypted data. In the first approach, data can be packed before become encrypted to prevent the additional cost of performing homomorphic operations. Only the data owner can pack clear data; thus, this approach may not be feasible in the settings where data owner has limited computation or storage resources. There are some applications where data need to be stored in a remote storage in a real-time fashion and in its original form. For example, in e-healthcare, it is necessary to capture vital signs such as heart rate in real-time and monitor its changes to detect any misbehavior of heart rate and predict the possibility of a heart attack in advance so it can be prevented. In this case, first data need to be encrypted and stored in remote storage. Then, remote storage provider can pack encrypted data when it helps to mitigate the cost of cryptographic operations.

There are also applications that allow data packing to be performed over clear data before data encryption. As an example, in the smart metering system, electricity consumptions that are captured in a regular timing fashion can be stored temporarily in the local database, then measurements can be packed together. Afterward, the packed measurement is encrypted and sent to the electricity service provider to calculate the bill, generate recommendations, statistics, etc.

REFERENCES

[1] N. P. Smart, *Cryptography Made Simple*, Information Security and Cryptography (Springer, 2016).

[2] O. Goldreich, *The Foundations of Cryptography - Volume 2, Basic Applications* (Cambridge University Press, 2004).

[3] B. Pinkas, T. Schneider, N. P. Smart, and S. C. Williams, *Secure two-party computation is practical,* in *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings* (2009) pp. 250–267.

[4] 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.

[5] I. Damgård, M. Geisler, and M. Krøigaard, *A correction to 'efficient and secure comparison for on-line auctions',* IJACT **1**, 323 (2009).

[6] J. Fan and F. Vercauteren, *Somewhat practical fully homomorphic encryption,* IACR Cryptology ePrint Archive **2012**, 144 (2012).

[7] 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.

[8] 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).

[9] 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.

[10] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, *Privacy-preserving face recognition,* in *Privacy Enhancing Technologies, 9th International Symposium, PETS 2009, Seattle, WA, USA, August 5-7, 2009. Proceedings* (2009) pp. 235–253.

[11] M. Barni, T. Bianchi, D. Catalano, M. D. Raimondo, R. D. Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, F. Scotti, and A. Piva, *Privacy-preserving fingercode authentication,* in *Multimedia and Security Workshop, MM&Sec 2010, Roma, Italy, September 9-10, 2010* (2010) pp. 231–240.

[12] M. Beye, Z. Erkin, and R. L. Lagendijk, *Efficient privacy preserving k-means clustering in a three-party setting,* in *2011 IEEE International Workshop on Information Forensics and Security, WIFS 2011, Iguacu Falls, Brazil, November 29 - December 2, 2011* (2011) pp. 1–6.

[13] A. Jeckmans, A. Peter, and P. Hartel, *Efficient privacy-enhanced familiarity-based recommender system,* in *Computer Security–ESORICS 2013* (Springer, 2013) pp. 400–417.

[14] C. Hazay and T. Toft, *Computationally secure pattern matching in the presence of malicious adversaries,* J. Cryptology **27**, 358 (2014).

[15] R. Cramer, E. Kiltz, and C. Padró, *A note on secure computation of the moore-penrose pseudoinverse and its application to secure linear algebra,* in *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings* (2007) pp. 613–630.

[16] 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.

[17] 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.

[18] J. R. Troncoso-Pastoriza, S. Katzenbeisser, M. U. Celik, and A. N. Lemma, *A secure multidimensional point inclusion protocol,* in *Proceedings of the 9th workshop on Multimedia & Security, MM&Sec 2007, Dallas, Texas, USA, September 20-21, 2007* (2007) pp. 109–120.

I.3 The Main Challenges in Developing Efficient and Secure e-Healthcare Systems

In this chapter, we introduce three different scenarios for developing realistic e-healthcare systems. These three scenarios are designed based on having detailed technical consultation with the experts in the field from both academia and industry. The scenarios are ordered from a realistic setting to more future design settings. The main goals of such systems are to achieve highly efficient and privacy-by-design e-healthcare systems. Considering the goals, we investigate the challenges of Securing the scenarios by using cryptographic techniques. The challenges include computation and communication costs and protecting privacy-sensitive data, while they are being processed.

3.1. E-HEALTHCARE SCENARIOS

In e-healthcare systems, there are settings that play important roles in the performance and security of the systems. Our scenarios, in Figures I.3.1, I.3.2, and I.3.3, are different in the following settings:

- Flow of data: It is the life-cycle of the data, where they are created, stored, and processed. It also states whether to centralize all the data or store them in a distributed form.
- Key management: This setting describes how the keys are distributed and used by parties. Moreover, it clarifies what keys are used by each party to encrypt before outsourcing data and what keys are used for encryption of data to be stored in the local storage.
- Collaborations: In the systems with multiple parties, it is important to clarify how parties are collaborating. There are several reasons for collaboration between parties such as joint computation, data storage and retrieval, key distribution, etc.
- Level of trust: Although we consider the semi-honest security setting, there are parties that are marked as trusted. For example, in e-healthcare systems, hospitals are trusted parties since they should have access to the patients' data. As another example, a research institute can be treated as a trusted or semi-trusted party in e-healthcare based on the system and security configurations. Choosing the level of trust on each party has a significant effect in choosing proper building blocks to make sure no party learns from the protocol more than his entitled to.

In this chapter, we state the objectives and investigate the challenges to achieve an efficient and secure e-healthcare system according to the desired settings in each scenario. Settings

- Centralized single-key based e-healthcare system (scenario 1): In this scenario, the private measurements are collected from the smart device service providers (DSPs) and stored in a centralized database storage (cloud). Since DSPs do no trust the cloud to access the measurements in clear, they encrypt their measurements by using the same public key before sending them to the cloud. Therefore, all the measurements are stored in a database and encrypted under the same key. In scenario 1, one of the parties is key manager, who generates a pair of public and private keys, and share the public key with the other parties.
- Centralized multiple-key based e-healthcare system (scenario 2): In scenario 2, similar to scenario 1, the measurements are stored in a centralized fashion in the cloud. However, in scenario 2, each DSPs uses its public key to encrypt the measurements before sending them to the cloud. Thus, in scenario 2, we remove the key manager from the system.
- Decentralized multiple-key based e-healthcare system (scenario 3): In scenario 3, unlike the scenarios 1 and 2, DSPs do not share their measurements with the cloud and keep them in their local storage. Moreover, DSPs use their public keys when collaborating with each other to process the measurements.

The system and security settings in the first scenario are based on the strong assumption of using single-key setting. The assumptions in scenario-1 result in more efficient e-healthcare system, but less secure than the other two scenarios. In contrast, the strong assumptions are removed from the third scenario, which results in a more computational and communicational demanding e-healthcare system with a higher level of security. In this thesis, we include the third scenario (scenario-3) for the purpose of completeness, but we do not address the challenges. The terms single-key and multiple-key refer to the key settings in communication channel between DSPs and the cloud.

3.2. CENTRALIZED SINGLE-KEY BASED E-HEALTHCARE SYSTEM (SCENARIO 1)

3.2.1. STAKE HOLDERS

There are five parties in the first scenario, as it is shown in Figure I.3.1, that are as follows:

- Patients: they are equipped with smart medical devices that can capture vital signals such as blood sugar, blood pressure, etc. Patients may purchase multiple medical devices from different device providers based on offered products and services. The devices capture the signals at regular intervals and send the measurements to their corresponding service providers.

- Smart Device Service Providers (DSPs): They offer different types of medical devices. Moreover, they are responsible for collecting the measurements from the patients and storing them in their local databases. Thus, DSPs enable patients to access their medical data from anywhere and anytime. DSPs also send a copy of the measurements to the cloud. Recall that DSPs are the owners of the cloud resources. DSPs have limited local computation and storage resources; thus, resource demanding operations cannot be performed in DSPs.

- Cloud: It has storage and computation resources and a service that communicates with DSPs, key manager, and medical institute. DSPs are the cloud service owners and cloud consumers. The cloud service collects the measurements from the DSPs and stores them in its local database. It also receives queries for different requests such as generating statistics and recommendation from medical institutes. Then, the cloud service processes the measurements in its local database according to the given queries. For simplicity, we will denote the cloud service as cloud in the rest of this thesis.

- Medical Institutes: Their primary goal is to improve public health through different phases:

    1. Monitoring and reporting patients' health conditions.
    2. Detecting any abnormal behavior of a patient's vital signal.
    3. Predicting the health condition of a patient based on his measurements, physical activity, diet, etc.

4. Generating statistics and customized recommendations to improve the health condition of patients and prevent diseases.

Medical institutes are trusted to retrieve all the measurements; however, they have limited computation and storage resources. Therefore, they communicate with the cloud in order to execute their queries.

- key Manager: This party generates a single pair of keys (private and public keys). Then, it shares the public key with other DSPs, cloud, and medical institute.

3.2.2. SECURITY AND SYSTEM SETTINGS

Security and system settings of the scenario-1 are as follows:

- In this scenario, captured measurements by medical devices from the DSPs are stored in the remote storage, cloud, via DSPs. A copy of the measurements is stored in the DSPs' local databases.
- DSPs and medical institutes are trusted parties in the system, while the cloud provider and key manager are semi-trusted.
- DSPs encrypt the measurements with the same public key, that is given from the key manager, before sending them to the cloud storage. Therefore, the ciphertexts in the cloud database are all encrypted under the same public key.
- Patients use different public keys given from their DSPs to encrypt their measurements.
- Medical institutes use the key manager's public key to encrypt their queries before sending them to the cloud. Recall that the cloud provider is semi-trusted .
- All parties except patients are in contact with the key manager for three reasons: 1) obtaining the public key, 2) joint computation, and 3) secure decryption.

Note that DSPs store the encrypted measurements under their public keys in their local storage. However, they send the encrypted version of the measurements under the key manager's public key, which is the same for all DSPs, to the cloud.

In this scenario, medical devices are measuring vital signals of patients and send them to the patients' mobile applications. The mobile applications encrypt the measurements with the public key of the corresponding DSP and send the ciphertexts to the DSP. DSPs store the given encrypted measurements in their local databases. They also send a copy of the measurements encrypted under the key manager's public key to the cloud. Cloud collects the encrypted measurements and stores them in its database. Since the cloud does not have the private key, it cannot decrypt the ciphertexts and learn about the privacy-sensitive medical information. The goals of the medical institute are obtaining statistics or measurements for one or a group of patients. Then, based on the received results from the cloud, medical institute asks the cloud to generate a customized recommendation for a patient. Since the medical institutes do not trust the cloud to learn about the query, they encrypt their queries with the key managers' public

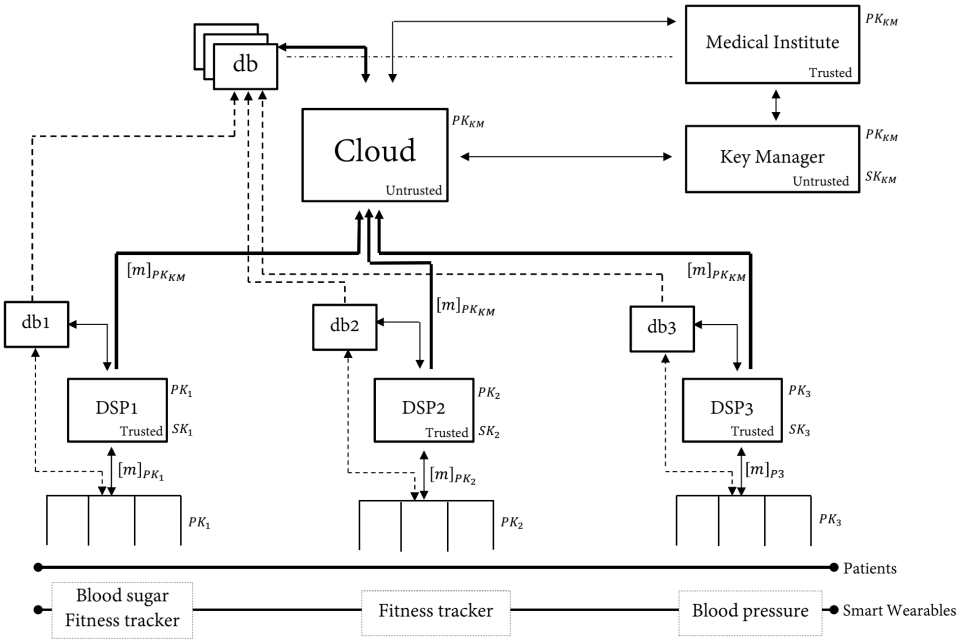

Figure I.3.1: e-Healthcare System: Scenario-1

key and sends the encrypted version of the query to the cloud. Then, the cloud and key manager perform one or multiple joint protocols to execute the query such that they do not learn any information about the query and the intermediate and final results.

3.2.3. CHALLENGES

- Data filtering: One of the main tasks of the cloud is to filter encrypted data. Data filtering is used in privacy-preserving data searching and retrieval, and generating recommendation and statistics. Filtering encrypted data necessitates using core building blocks such as comparison and equality testing protocols. Let us consider two examples: first; a medical institute is interested in the number of patients having blood pressure and blood glucose above specific thresholds in a specific period. As another example, a query may ask for computing the average of blood glucose of patients above the desired age between 2:00 pm and 4:00 pm, when most of the people already had their launch, in the last 60 days. This query contains three AND operations, three comparisons, and one equality testing. Considering the performance of state-of-the-art core building blocks, executing such queries over a large set of encrypted measurements takes a very long time. Moreover, using existing building-blocks results in a significant amount of communication cost.

- Data retrieving: Considering a database which its size is growing every second, retrieving data of a group of patients is a challenging task when data are encrypted.

In the case of using an index to find and retrieve encrypted data, its security and performance of searching and updating are the challenges. Assuming that both cloud and key manager are semi-trusted, the two parties perform joint cryptographic protocols, preserving the privacy of the measurements in index-based data searching and retrieval is a must. Moreover, since the cloud is collecting encrypted data from multiple DSPs and each measurement has multiple attributes, the performance of updating the index in terms of computation and communication is also a challenge.

- Medical recommendations: A recommender system can generate recommendations for patients to improve their health conditions. As one example, a recommender system can obtain information about how much running in what time of day has positive effects on blood pressure based on the given statistics and then generate recommendations for patients accordingly. As another example, a recommender system may ask the cloud to find all those patients that are more successful in balancing their blood sugar levels than others. Then, it investigates in their diets, physical activities, sleeping patterns, etc. to learn their lifestyles. Afterward, the result can be used to generate customized recommendations for other patients. From a security viewpoint, during the process of searching, learning the lifestyle, and generating the recommendation, it is vital to preserving the confidentiality of data. From the efficiency point of view, generating recommendation by using the existing cryptographic solutions demands a significant amount of computation and communication resources.

3.3. CENTRALIZED MULTIPLE-KEY BASED E-HEALTHCARE SYSTEM (SCENARIO 2)

In the first scenario two strong assumptions are:

- All DSPs use the same public key of key manager when encrypting the measurements before sending them to the cloud.
- There is an added party called the key manager who is trusted to follow the protocol instructions and do not change, remove, or add any data.

In scenario-2, these two assumptions are changed to achieve a more secure e-healthcare system. In scenario-2, DSPs send the measurements that are encrypted under their public keys to the cloud and the key manager is removed from the system.

3.3.1. STAKE HOLDERS

As it is shown in Figure I.3.2, there are four parties in the second scenario.

- Patients: Similar to the scenario-1, patients use their medical devices and the mobile application to capture their vital signs and send them to their DSPs in encrypted form.

- DSPs: They collect the measurements from patients and store them in their local databases. However, unlike the scenario-1, DSPs send a copy of the measurements to the cloud in encrypted form under their public keys.

- Cloud: It collects and stores encrypted measurements received from DSPs in its local database. Cloud has only access to the public keys of the measurements in its storage; thus, it has to communicate with DSPs to perform cryptographic protocols that require the private keys.

- Medical Institutes: Similar to the scenario-1, they are sending queries to the cloud.

3.3.2. SECURITY AND SYSTEM SETTINGS

Security and system settings of the scenario-2 are as follows:

- DSPs and medical institutes are trusted parties in the system, but the cloud is semi-trusted.

- All the measurements are centralized in the cloud database.

- DSPs send a copy of their measurements under their public keys to the cloud. Therefore, the cloud has a database of measurements encrypted under different public keys.

- Patients use different public keys given from their DSPs to encrypt their measurements.

- Medical institutes use their public keys to encrypt their queries.

3.3.3. CHALLENGES

Although scenario-2 has advantages comparing to the scenario-1, removing the key manager and allowing DSPs to send their measurements encrypted under their public keys introduce new challenges. The scenario-2 includes all the challenges of scenario-1 plus more challenges that are described in bellow:

- DSPs have only limited computational resources; therefore, the joint cryptographic protocols between the cloud and DSPs should be designed such that most of the computations are performed in the cloud. Moreover, the system should be designed such that it minimizes the communication round and data transmission complexities between the cloud and DSPs.

- Cloud has a database of measurements that are encrypted under different public keys. In this setting, performing operations over ciphertexts such as generating statistics are not as straightforward as the case in scenario-1. To process encrypted measurements under different keys, first, cloud has to convert the ciphertexts to new ciphertexts under the same public key.

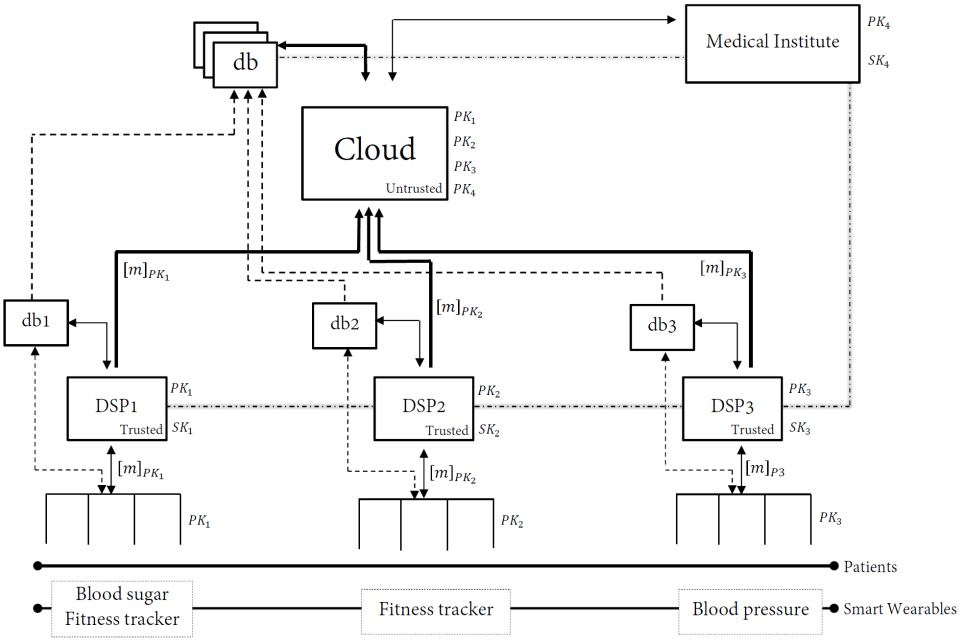

Figure I.3.2: e-Healthcare System: Scenario-2

3.4. DECENTRALIZED MULTIPLE-KEY BASED E-HEALTHCARE SYSTEM (SCENARIO 3)

In the first two scenarios, we construct the e-healthcare system in a centralized form. Centralizing data makes the data more accessible. Considering the cloud as data centralizer, it can carry out resource demanding operations over encrypted data. In the third scenario, we assume that the data are stored in a decentralized fashion. The data are stored only in DSPs' local databases. Moreover, we do not assume the medical institutes to be fully trusted to access to all the data, but it can execute queries for generating statistics or recommendations. The primary motivation for assuming this scenario is to investigate the practicability of developing secure e-healthcare systems with less trust and the number of parties.

3.4.1. STAKE HOLDERS

As it is shown in Figure I.3.3, there are three parties in the third scenario.

- Patients: Similar to the other two scenarios, patients outsource their measurements into the DSPs via a mobile application.
- DSPs: They collect the measurements from patients and store them in their local databases in encrypted form under their public keys.
- Medical institutes: They are collaborating with DSPs to execute their queries. Med-

ical institutes are only interested in receiving statistics or generating recommendations.

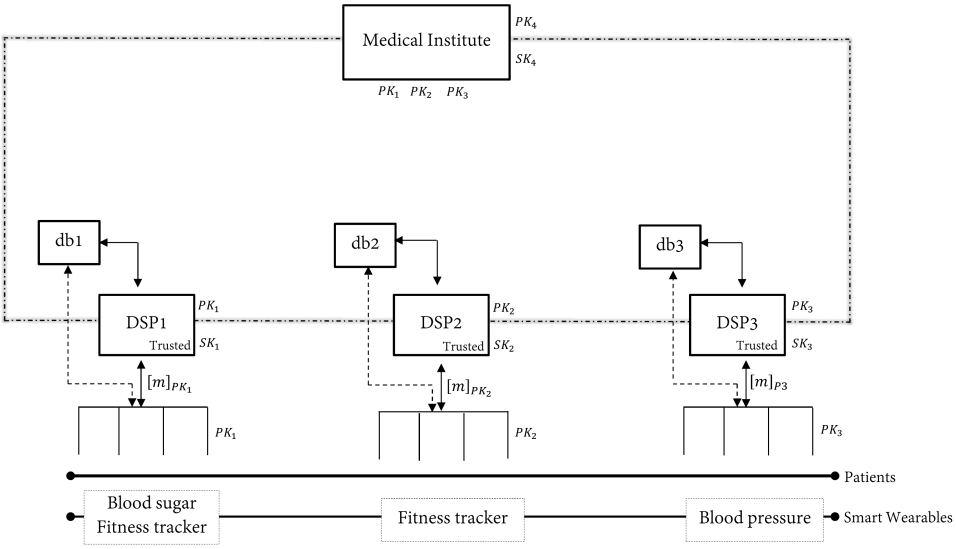

Figure I.3.3: e-Healthcare System: Scenario-3

3.4.2. SECURITY AND SYSTEM SETTINGS

Security and system settings of the scenario-3 are as follows:

- DSPs are trusted parties and have access to their customers' measurements in clear.
- Medical institutes are not fully trusted to access patients' data, but they can execute queries for generating statistics and recommendations.
- To execute given queries from medical institutes, DSPs can collaborate directly or via medical institutes.

3.4.3. CHALLENGES

Achieving a privacy-preserving e-healthcare system might be more promising in the third scenario than the other two scenarios. The main reason is that the confidential data are retained in the DSPs and no other semi-trusted parties have access to the measurements in clear or encrypted form. However, the setting introduces new challenges that are described as follows:

- Both medical institutes and DSPs have limited computational resources. This limitation makes processing encrypted data a challenging task for both sides.

- Since the medical institutes are semi-trusted, it is necessary to make sure they cannot learn about patients' private data from the generated statistics or recommendations.
- The cryptographic protocols should be designed such that they are compatible with processing encrypted data stored in multiple storages.

I.4 | Existing Applications of Cryptographic building blocks

In this chapter, we investigate different applications that are using core cryptographic building blocks, namely comparison, equality testing, and data packing, for processing confidential data. We analyze the contribution of building blocks in the total computation and communication complexities of the secure versions of the applications. The complexities are provided for the papers which reported a clear performance analysis; otherwise, the complexities are estimated if enough information is given in their works. Since there are limited publications in the e-healthcare domain that are using homomorphic encryption for private data processing, we also investigate applications beyond e-healthcare to obtain better results.

4.1. INTRODUCTION

In chapter I.3, we designed three realistic and secure e-healthcare scenarios. Then, for each scenario, we explained the challenges according to the system and security settings. We showed that there are building blocks that play key roles in our e-healthcare scenarios such as comparison, equality testing, data filtering, and data packing. In this chapter, we analyze the performance of several applications that are using the building blocks which are used in our scenarios. We show that how much of the total required resources, computation and communication, for performing the applications are used by the building blocks. In our analysis, we also check whether data packing is used in the applications. The result of this chapter helps to see whether improving the existing building blocks and techniques to reduce the number of homomorphic operations are effective to reduce the total costs of the applications significantly.

4.2. EXISTING APPLICATIONS OF CRYPTOGRAPHIC BUILDING BLOCKS

In this section, we choose papers from various research fields such as recommender systems, pattern recognition, and data classification. For each paper, we give a brief description of its application, and then, we discuss the building blocks that are used.

4.2.1. MEDICAL RECOMMENDATION AND PREDICTION SYSTEMS

In this section, we investigate two application domains, namely medical recommender and disease prediction systems, which rely on the cryptographic building blocks. We briefly explain those medical services and the ways to preserve the confidentiality of medical data while they are being processed.

MEDICAL RECOMMENDER SYSTEM

One of the challenging task for the patients is to find the best physicians based on their health conditions. In [1], Ryan Hoens et al. introduced a system for patients to obtain a list of best physicians in a ranked form. In their system, the privacy of sensitive information from both patients and physicians are preserved. Ryan Hoens et al. used a homomorphic data encryption [2] to protect confidential data, while data can be processed securely. The medical recommender system sorts the list of candidate physicians that are chosen based on their encrypted scores according to the patients' diseases. The scores are computed from the experience of the physicians and feedback given by patients who already visited the physicians.

To rank encrypted scores, first, the algorithm (BITS) from [3] used to extract ℓ bits of encrypted scores. Then, the comparison protocol (BIT-LE) introduced in [4] is used to rank the encrypted ℓ-bit scores. BIT-LE requires $110\ell \log_2 \ell + 140\ell$ executions of the MULT protocol [5]. The data transmission costs are $O(\ell n^2 \kappa)$ bits for BITS, $O(n^2 \kappa^2 \log \kappa)$ for the bit decomposition, $O(n\kappa)$ for the MULT. Table I.4.1 shows the computation and data transmission complexities of [1]. It also shows that what percent of the complexity is dominated by the core building blocks comparison and equality testing protocols. Table I.4.1 also shows whether the data packing is used in the protocol.

DISEASE RISK PREDICTION

Nowadays, genomic data are being widely used by medical institutes to analyze and predict the risk of diseases. However, genomic data includes privacy-sensitive information, which raises privacy concerns among patients. To fill the privacy gap, Ayday et al. [6] propose a system to process genomic data and predict disease risk in a privacy-preserving manner. They used the comparison protocol in [7] to compute the genetic regression coefficient that is used to obtain the final disease risk. According to the experimental results provided in [6], comparison protocol for 16 bits inputs takes more than one second, wherein the original work [7] it takes only 40 milliseconds. Although the comparison protocol is called only three times in [6] and its run-time takes just a small fraction of the total run-time, the efficiency of their system would significantly decrease if the comparison protocol has been called more. Later, duVerle et al. [8] introduced a new approach to achieving a privacy-preserving genome sequencing and analyzing system by using homomorphic encryption. They developed an Oblivious Comparison to Zero, that is called GreaterThanZero. The communication and the computation complexities of GreaterThanZero is $O(\log \ell)$. GreaterThanZero communication and computation costs are taking 39% and 30% of the total communication and computation costs, respectively.

Table I.4.1: Contribution (%) of the core building blocks in total computation (Comp.) and total data transmission (DT) costs of related work. In the table, ℓ is the bit-length of inputs, κ is the security parameter for asymmetric encryption, t is the security parameter for symmetric encryption, and DP shows whether data packing is used. Explanation regarding x and K is provided in [9]. The complexities that are filled with (–) shows that the work lacks complexity analysis.

Application	Comp.	(%)	DT	(%)	DP
Medical recommender[1]	$O(\ell \log \ell)$	75	$O(\ell n \kappa)$	80	no
Genome analyzing[8]	$O(\log \ell)$	30	$O(\log \ell)$	39	yes
Location proximity[10]	$O(2^{\ell})$	90	$O(2^{\ell})$	99	no
Face recognition[7]	$O(\ell^2)$	75	$O(\ell)$	63	no
Iris identification[11]	–	72	$O(t\ell)$	9	no
Fingerprint identification[11]	–	94	$O(t\ell)$	85	no
Graph matching[12]	$O(\sqrt{\ell}(\kappa + \log \ell))$	–	$O(\log \ell \log^* \ell)$	–	no
Graph matching[13]	$O(\ell^2)$	99	$O(\ell)$	79	yes
Data classification[14]	$O(\ell^2)$	83	$O(\ell)$	77	no
ECG Classification(LBP)[15]	$O(t)$	95	$O(\kappa + t)$	60.5	yes
ECG Classification(NN)[15]	$O(1)$	100	$O(1)$	100	yes
User clustering[16]	$O(\ell^2)$	21.5	$O(\ell)$	13.5	no
Fingercode authentication[17]	–	63	$O(\ell t + \kappa)$	54	yes
Private database access[18]	$O(\ell \log \ell)$	59	$O(\ell \log \ell)$	–	yes
Encryption switching[19]	–	–	$O(\ell)$	75	no
Smart metering[20]	$O(\ell)$	50	–	–	no
Car access provision[21]	$O(\ell)$	67	–	–	no
Software analysis[9]	$O(\ell)$	71	$O(\ell)$	83	yes

4.2.2. PATTERN RECOGNITION

Pattern recognition techniques have been used in many applications such as biometric recognition and authentication, graph matching, and profile matching. In this section, we briefly explain a few applications of pattern recognition, then we show how existing works address the privacy issues and what building blocks they have used in their solutions.

BIOMETRIC RECOGNITION

In [7], Erkin et al. introduced a privacy-preserving face recognition system. In their system, there are two parties where the first party has access to the database of all facial templates and the second party has a face image. The challenge is to check whether the provided face image by the second party exists in the database of the first party such that at the end of the protocol both parties do not learn information about each others' data and the first party should learn the result of the protocol. Erkin et al. developed A secure comparison in their solution to compare two encrypted integers based on the technique proposed in [22]. In [22], a comparison protocol is designed for the setting where there are two parties and each holding a private integer in clear. The developed comparison protocol in [7] has the computation complexity of $\ell(\ell+3)/2$ homomorphic exponentiations, $\ell(\ell+1)$ homomorphic multiplications, and ℓ DGK zero check [22] for a single comparison. More details about the performance of the introduced comparison protocol provided in Table I.4.1.

BIOMETRIC AUTHENTICATION

Technological developments in biometric scanners, biometric authentication based on fingerprint, iris, facial recognition are becoming more practical and reliable as a replacement for traditional techniques such as using passwords. Biometric authentication has two phases: 1) storing biometric template and 2) searching for desired biometric. One of the widely used methods of identification relies on the fingerprint-based solution. However, since fingerprint-based identification information is uniquely linked with people, there are privacy concerns on abusing biometric data stored in government agencies or companies. In [17], a privacy-preserving fingerprint identification based on using homomorphic encryption is introduced. In their scenario, the client has a fingerprint reader device and wants to check if the fingerprint scanned by the device matches any fingerprint template stored in the server in a privacy-preserving form. The solution uses a cryptographic tool called bit-MIN that relies on a secure comparison protocol based on garbled circuits for finding the minimum.

Similarly, a secure fingerprint recognition system and a secure protocol for iris and fingerprint identification are introduced in [23] and [11], respectively. Blanton and Gasti in [11], addressed the problem of biometric identification using two types of biometrics, Iris and fingerprint. Blanton and Gasti used homomorphic data encryption and the building blocks such as secure comparison to make a security-by-design biometric authentication. They used garbled circuits based comparison protocol from [24] in their systems. The computation complexity of the used comparison protocol is roughly ℓ non-XOR 2-to-1 gates and ℓ times invoking SHA265 hash function. The data transmission complexity of the comparison protocol is $O(t\ell)$, where t is the security parameter for symmetric encryption.

Computing the graph edit distance is one of the techniques used for pattern recognition and biometric identification. Graph edit distance measure the difference between two graphs, which is also used in graph exact and error-tolerant matching. Mandal et al. in [12] introduced a privacy-preserving protocol to compute the graph edit distance between two graphs that are located in different parties. To achieve such protocol, Mandal et al. used a threshold additively homomorphic encryption scheme [25]. Moreover, a number of building blocks such as comparison and equality testing protocols are used in [12]. Mandal et al. also developed a building block to find the maximum encrypted value (PMC) within an encrypted set based on the comparison protocol [26]. Toft in [26] introduced two comparison protocols in [26], one log round and one constant round protocols. The first protocol has $O(\log \ell)$ communication round and $O(\kappa \log \ell)$ computation complexities and the second protocol has $O(1)$ communication round and $O(\sqrt{\ell}(\kappa + \log \ell))$ computation complexities. Considering the complexity of the comparison protocol, both computation and communication complexities of PMC are stated as $O(n^2 + n\ell \log \ell \log^* \ell)$, where n is the size of the encrypted set.

GRAPH MATCHING FRAMEWORK

In [13], Chu and Chang propose a privacy-preserving protocol to find the best matching multimedia data that are encrypted and stored in remote storage by using homomorphic encryption. They used their solution in two applications: 1) video tag suggestion and 2) video copy detection. Several building blocks are used in their solutions such as comparison and zero-checking[7, 22, 24]. The complexities of the building blocks are discussed in previous applications. Table I.4.1 shows that the building blocks dominate 99% of the overall computation cost of the matching protocol.

4.2.3. CLASSIFICATION AND CLUSTERING

DATA CLASSIFICATION

Using efficient privacy-preserving building blocks play an important role in the total performance of secure data classifiers. Bost et al. [14] developed three cryptographic building blocks that are used in most of the classifiers: 1) comparison, 2) argmax , and 3) dot product, where the argmax task is to find the maximum in a set of encrypted integers by using comparison protocol. Assuming the set has t encrypted ℓ-bit integers, the cost argmax equals to performing $O(t)$ comparison protocol for the ℓ-bit inputs and $O(t)$ homomorphic linear operations such as multiplication and exponentiation. The core of the developed comparison protocol in [14] is similar to the protocol in [7]. After developing the building blocks, their efficiencies are evaluated by applying them on several well-known classifiers. As it is stated in their evaluation results, the comparison and argmax protocols take 90% and 83% of the total computation costs of linear and Naïve Bayes Classifiers, respectively.

Barni et al. [15, 27] secured two ECG classification techniques based on LBP [28] and NN [29]. To achieve more efficient systems, they used building blocks that are based on the combination of garbled circuits [24] and homomorphic encryption [30]. Since performing a large number of homomorphic operation is expensive they also apply data packing to improve the efficiency. Apart from the costs of homomorphic operations, data packing helps to improve the efficiency of the decryption process, because a pack

of multiple encrypted values can be decrypted at the cost of decrypting one ciphertext. The total computation cost of securing LBP includes $3t + 1233$ evaluations of garbled gates and $2t + 1$ homomorphic exponentiations, and the communication cost is $6t^2 + 3735t + 32\kappa$ bits. The computation complexity of securing NN is 85156 evaluations of garbled circuits and 209 homomorphic exponentiations, and its communication cost is $18316t$ bits.

DATA CLUSTERING

Erkin et al. [16] introduced a privacy-preserving solution for the problem of grouping multiple users into different clusters based on their preferences. The K-means clustering algorithm is used for grouping users, which requires finding the minimum of a set of encrypted integers. To find the minimum securely, they used the secure comparison protocol in [7]. The number of times that the comparison protocol is called is equal to the number of clusters in an application. The communication cost of their solution is $O(KR)$, where K and R are the number of clusters and dimensional space of users, respectively. The overall computation cost of the clustering protocol is $O(K(\ell^2 + RM))$, where M and ℓ are the number of users and the bit length of the inputs of the comparison protocol.

LOCATION PROXIMITY

Tracking the objects and approximating the distance between two or multiple of them, known as Location Based Services (LBS), is becoming more popular. Applications of LBSs are tracking cargo throughout the sea, tracking a lost phone, and querying for a restaurant that is nearby. However, leakage of location-based information to unauthorized data receivers lead to privacy risks. To bridge the privacy gaps, Hallgren et al. [10] introduced a protocol that is called InnerCircle. InnerCircle uses a secure comparison protocol to proximate the location of objects in the privacy-preserving form. The comparison protocol used in [10] for the inputs $([x], [y])$ works as follows:

1. For $i \in \{0, \cdots, y-1\}$ computes $[a_i] = [x - i] = [x] \cdot [i]^{-1}$
2. Multiplicatively randomizes all the $[a_i]$ values and then shuffle them.
3. Sends all the $[a_i]$ values to the key manager.
4. Key manager decrypts all the $[a_i]$ values. If none of them are zero sends back $[1]$, otherwise, $[0]$.

The comparison protocol outputs $[1]$ if $x \geq y$, otherwise, $[0]$. The comparison protocol needs to compute y homomorphic subtractions and y decryption operations. In total, the comparison protocol dominates above 90% of the total computation cost.

4.2.4. OTHER APPLICATIONS

DATABASE ACCESS

Although storing and processing data in remote storage and computation servers are beneficial for individuals and companies, privacy issues regarding confidentiality of sensitive data is still an issue. There are different techniques to fill that privacy issue such as

data anonymization and data encryption. Data encryption as a well-known solution can guarantee data confidentiality; however, accessing encrypted data in an efficient and privacy-preserving way becomes a challenge. In [18], Gentry et al. presented a protocol for private queries that supports accessing the database while the data confidentiality is preserved. They used a somewhat homomorphic encryption scheme to develop a secure equality testing protocol. Then, they used their equality testing protocol to build a comparison protocol, which is the main building block of their ORAM. The equality testing protocol has $O(\log \ell)$ communication rounds and $O(\ell \log \ell)$ data transmission complexity.

The issue of achieving a data access protocol for remotely stored privacy-sensitive data such that the access pattern is hidden has addressed with different solutions. Techniques such as private information retrieval and oblivious RAM are among the solutions that address that issue. However, retrieving a vector of n elements using those two approaches might be computationally very expensive for both server and client. As a solution, a private lookup protocol is proposed in [31] to mitigate the computational cost of retrieving a vector of elements from remote storage such that the access pattern is still hidden. To achieve the lookup protocol a secure comparison and equality testing are used to boost up the efficiency.

DATA SEARCHING

Data filtering is an important tool in signal processing for searching within data that are stored in remote storage. Since the main protocols are mainly performed by the third party it is essential to make sure private data are kept confidential during storing, retrieving, processing data. Troncoso-Pastoriza and Pérez-González in [32] presented a privacy-preserving protocol to enable filtering remotely stored encrypted data by an untrusted remote storage provider. Data packing is also applied to decrease the computational costs of performing homomorphic operations.

DATA AGGREGATION

Mustafa et al. in [20] present a protocol for collecting customers' electricity consumption by electricity provider or SG provider. Their solution keeps confidentiality of private data while enables processing encrypted data by using homomorphic encryption. They also developed an equality testing protocol for bit-wise encrypted inputs with $O(\ell)$ communication round complexity, which demands ℓ secure multiplications.

ACCESS CONTROL

Unlike traditional car sharing and rental methods, Keyless Car Sharing Systems are becoming a research topic by companies like Volvo and BMW. KSS allows the owner of a car or car rental company easily transfer a digital token which temporary lets another driver to use the car. Although KSS is a more efficient and easy way of car sharing, it may leak location information of the drivers, since it is necessary to use KSS with GPS enabled. To mitigate such privacy issue, authors in [21] introduce a privacy-preserving decentralized car sharing system by using homomorphic data encryption and building-blocks such as secure equality testing protocol.

SOFTWARE ANALYSIS

As software becomes bigger and more complicated, their maintenance, updating, and upgrading become more sophisticated tasks. Software validation is one of those tasks that its complexity and difficulty grow with the size and complexity of the software, which makes software validation a challenging task. To address that challenge, new methods are introduced to monitor and improve the software in an online and real-time form. Collected information from the software while they are working, log files, play an important role in real-time monitoring. Moreover, since the size of the log files for big software are large, remote computation servers are used to boost up the efficiency of the whole software validation process. Tillem et al. in [9] introduce a privacy-preserving technique to analyze the log files while their confidentiality is preserved. In their solution, they used homomorphic data encryption to enable processing the encrypted log files. They also used building blocks such as secure equality testing protocol and data packing in their protocol that resulted in cutting down the communication cost by 83% and computation cost by 71%.

RECOMMENDER SYSTEM

Processing personal data and generating recommendation are challenging tasks if the data should be maintained confidential. Erkin et al. [33] introduce a privacy reserving recommender system. Then, they apply data packing on their solution, and then present its performance gain. According to the implementation results in [33], the data transmission cost of their recommender system for a user when data packing is not applied is 495 KB. However, after packing the ciphertexts, the data transmission cost decreased to 11.5 KB that is 98% improvement. Using data packing on encrypted data has also significantly decreased the computation cost of the recommender system from 50 hours to 38 minutes that is 99% improvement. Erkin et al. also applied the data packing on the comparison protocol, which decreased the computation cost of comparison protocol in the recommender system from 2 minutes to 35 seconds.

Apart from the applications above, we list other applications that also used the core building blocks.

- **Image Feature Extraction:** In [34], the Challenges of performing a privacy-preserving form of scale-invariant feature transform is studied.
- **Evaluating Decision Trees and Random Forests:** One of the well-known types of classifiers is decision trees and random forests. In [35], authors introduce a decision tree evaluation protocol that preserves the data confidentiality.
- **Processing Floating Point Signals:** [36] presented an implementation of IEEE 754 floating point standard, which helps to process encrypted non-integer numbers with a floating number.
- **Watch List Screening in Video Surveillance System:** In [37], a watch list screening system is proposed that enables identifying people such that the result of identification process does not leak any private information to unauthorized people.
- **Integer Division:** In [38], an encrypted integer division protocols based on the comparison protocol is proposed.

- **Similarity Evaluation of Time Series Data:** In [39], an efficient and secure protocol for computing the similarity over encrypted time series data is presented.
- **Secure Sorting:** In [40], authors introduced an equality testing build block to make an efficient sorting protocol.

4.3. CONCLUSION

In this chapter, we analyzed the performance the cryptographic building blocks in several applications, which is summarized in Table I.4.1. It is clear that the core building blocks, namely comparison and equality testing, are dominating the computation and communication costs of the most of the applications. Therefore, developing efficient cryptographic core building blocks helps to improve the performance of the applications significantly. Moreover, according to the analysis in [33], we can conclude that developing techniques such as data packing, which reduce the number of homomorphic operations in the building blocks can be very effective to mitigate the costs of the applications.

REFERENCES

[1] T. R. Hoens, M. Blanton, A. Steele, and N. V. Chawla, *Reliable medical recommendation systems with patient privacy,* ACM TIST **4**, 67:1 (2013).

[2] 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.

[3] B. Schoenmakers and P. Tuyls, *Efficient binary conversion for paillier encrypted values,* in *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings* (2006) pp. 522–537.

[4] I. Damgård, M. Fitzi, E. Kiltz, J. B. Nielsen, and T. Toft, *Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation,* in *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings* (2006) pp. 285–304.

[5] R. Cramer, I. Damgård, and U. M. Maurer, *General secure multi-party computation from any linear secret-sharing scheme,* in *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding* (2000) pp. 316–334.

[6] E. Ayday, J. L. Raisaro, P. J. McLaren, J. Fellay, and J. Hubaux, *Privacy-preserving computation of disease risk by using genomic, clinical, and environmental data,* in *2013 USENIX Workshop on Health Information Technologies, HealthTech '13, Washington, DC, USA, August 12, 2013* (2013).

[7] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, *Privacy-preserving face recognition,* in *Privacy Enhancing Technologies, 9th International Symposium, PETS 2009, Seattle, WA, USA, August 5-7, 2009. Proceedings* (2009) pp. 235–253.

[8] D. A. duVerle, S. Kawasaki, Y. Yamada, J. Sakuma, and K. Tsuda, *Privacy-preserving statistical analysis by exact logistic regression,* in *2015 IEEE Symposium on Security and Privacy Workshops, SPW 2015, San Jose, CA, USA, May 21-22, 2015* (2015) pp. 7–16.

[9] 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.

[10] P. A. Hallgren, M. Ochoa, and A. Sabelfeld, *Innercircle: A parallelizable decentralized privacy-preserving location proximity protocol,* in *13th Annual Conference on Privacy, Security and Trust, PST 2015, Izmir, Turkey, July 21-23, 2015* (2015) pp. 1–6.

[11] M. Blanton and P. Gasti, *Secure and efficient protocols for iris and fingerprint identification,* in *Computer Security - ESORICS 2011 - 16th European Symposium on Research in Computer Security, Leuven, Belgium, September 12-14, 2011. Proceedings* (2011) pp. 190–209.

[12] K. Mandal, B. Alomair, and R. Poovendran, *Secure error-tolerant graph matching protocols,* in *Cryptology and Network Security - 15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings* (2016) pp. 265–283.

[13] W. Chu and F. Chang, *A privacy-preserving bipartite graph matching framework for multimedia analysis and retrieval,* in *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval, Shanghai, China, June 23-26, 2015* (2015) pp. 243–250.

[14] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser, *Machine learning classification over encrypted data,* in *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015* (2015).

[15] M. Barni, P. Failla, R. Lazzeretti, A. Sadeghi, and T. Schneider, *Privacy-preserving ECG classification with branching programs and neural networks,* IEEE Trans. Information Forensics and Security **6**, 452 (2011).

[16] Z. Erkin, T. Veugen, T. Toft, and R. L. Lagendijk, *Privacy-preserving user clustering in a social network,* in *First IEEE International Workshop on Information Forensics and Security, WIFS 2009, London, UK, December 6-9, 2009* (2009) pp. 96–100.

[17] M. Barni, T. Bianchi, D. Catalano, M. D. Raimondo, R. D. Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, F. Scotti, and A. Piva, *Privacy-preserving fingercode authentication,* in *Multimedia and Security Workshop, MM&Sec 2010, Roma, Italy, September 9-10, 2010* (2010) pp. 231–240.

[18] C. Gentry, S. Halevi, C. S. Jutla, and M. Raykova, *Private database access with he-over-oram architecture,* in *Applied Cryptography and Network Security - 13th International Conference, ACNS 2015, New York, NY, USA, June 2-5, 2015, Revised Selected Papers* (2015) pp. 172–191.

[19] 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.

[20] M. A. Mustafa, S. Cleemput, A. Aly, and A. Abidin, *An mpc-based protocol for secure and privacy-preserving smart metering,* in *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe, ISGT-Europe 2017, Torino, Italy, September 26-29, 2017* (2017) pp. 1–6.

[21] I. Symeonidis, A. Aly, M. A. Mustafa, B. Mennink, S. Dhooghe, and B. Preneel, *Sepcar: A secure and privacy-enhancing protocol for car access provision,* in *Computer Security - ESORICS 2017 - 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part II* (2017) pp. 475–493.

[22] 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.

[23] M. Barni, T. Bianchi, D. Catalano, M. D. Raimondo, R. D. Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, A. Piva, and F. Scotti, *A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingercode templates,* in *Fourth IEEE International Conference on Biometrics: Theory Applications and Systems, BTAS 2010, Washington, DC, USA, 27-29 September, 2010* (2010) pp. 1–7.

[24] V. Kolesnikov, A. Sadeghi, and T. Schneider, *Improved garbled circuit building blocks and applications to auctions and computing minima,* in *Cryptology and Network Security, 8th International Conference, CANS 2009, Kanazawa, Japan, December 12-14, 2009. Proceedings* (2009) pp. 1–20.

[25] C. Hazay, G. L. Mikkelsen, T. Rabin, and T. Toft, *Efficient RSA key generation and threshold paillier in the two-party setting,* in *Topics in Cryptology - CT-RSA 2012 - The Cryptographers' Track at the RSA Conference 2012, San Francisco, CA, USA, February 27 - March 2, 2012. Proceedings* (2012) pp. 313–331.

[26] 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* (2011) pp. 174–191.

[27] R. Lazzeretti and M. Barni, *Privacy preserving classification of ECG signals in mobile e-health applications,* in *Medical Data Privacy Handbook* (2015) pp. 569–611.

[28] D. Ge, N. Srinivasan, and S. M. Krishnan, *The application of autoregressive modeling in cardiac arrhythmia classification,* in *Advances in cardiac signal processing* (Springer, 2007) pp. 209–226.

[29] D. Ge, N. Srinivasan, and S. M. Krishnan, *Cardiac arrhythmia classification using autoregressive modeling,* Biomedical engineering online **1**, 5 (2002).

[30] B. Pinkas, T. Schneider, N. P. Smart, and S. C. Williams, *Secure two-party computation is practical,* in *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings* (2009) pp. 250–267.

[31] P. Laud, *A private lookup protocol with low online complexity for secure multiparty computation,* in *Information and Communications Security - 16th International Conference, ICICS 2014, Hong Kong, China, December 16-17, 2014, Revised Selected Papers* (2014) pp. 143–157.

[32] J. R. Troncoso-Pastoriza and F. Pérez-González, *Secure adaptive filtering,* IEEE Trans. Information Forensics and Security **6**, 469 (2011).

[33] 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).

[34] C. Hsu, C. Lu, and S. Pei, *Image feature extraction in encrypted domain with privacy-preserving SIFT,* IEEE Trans. Image Processing **21**, 4593 (2012).

[35] D. J. Wu, T. Feng, M. Naehrig, and K. E. Lauter, *Privately evaluating decision trees and random forests,* PoPETs **2016**, 335 (2016).

[36] M. Franz and S. Katzenbeisser, *Processing encrypted floating point signals,* in *Proceedings of the thirteenth ACM multimedia workshop on Multimedia and security* (ACM, 2011) pp. 103–108.

[37] H. Sohn, K. N. Plataniotis, and Y. M. Ro, *Privacy-preserving watch list screening in video surveillance system,* in *Advances in Multimedia Information Processing - PCM 2010 - 11th Pacific Rim Conference on Multimedia, Shanghai, China, September 21-24, 2010, Proceedings, Part I* (2010) pp. 622–632.

[38] C. Ugwuoke, Z. Erkin, and R. L. Lagendijk, *Secure fixed-point division for homomorphically encrypted operands,* in *Proceedings of the 13th International Conference on Availability, Reliability and Security, ARES 2018, Hamburg, Germany, August 27-30, 2018* (2018) pp. 33:1–33:10.

[39] H. Zhu, X. Meng, and G. Kollios, *Privacy preserving similarity evaluation of time series data,* in *Proceedings of the 17th International Conference on Extending Database Technology, EDBT 2014, Athens, Greece, March 24-28, 2014.* (2014) pp. 499–510.

[40] K. V. Jónsson, G. Kreitz, and M. Uddin, *Secure multi-party sorting and applications,* IACR Cryptology ePrint Archive **2011**, 122 (2011).

II

Secure Equality Testing Protocols

II.1 | Efficient and Secure Equality Tests

Abstract

Secure equality testing of two private values is one of the fundamental building blocks of many cryptographic protocols designed for Signal Processing in the Encrypted Domain (SPED). Existing protocols introduce significant amount of computation and computational overhead, which makes it essential to search for new and novel, efficient equality tests for the design of SPED algorithms. In this paper, we first describe the state-of-the-art equality tests, and then propose two cryptographic protocols which are significantly more efficient than the existing work. Our proposals achieve high performance due to algorithmic changes and successful deployment of data packing. Furthermore, we also present a novel secure exponentiation protocol as a part of our first equality test. Complexity and performance analyses clearly indicate the high efficiency of our protocols in terms of computation cost.

This chapter has been published as "Efficient and Secure Equality Tests", by M.Nateghizad, Z.Erkin and R.L.Lagendijk in the proceedings of 8th *International Workshop on Information Forensics and Security*, 2016.

1.1. INTRODUCTION

RECENT advances in the collection, processing, and delivery of digital contents have been deployed in many domains. However, processing sensitive information have also raised several privacy concerns. Possible misuse or leakage of privacy-sensitive data has several consequences. Therefore, privacy research has been one of the most attractive topics in the last few years. SPED, as one of the solutions to preserve the privacy of users, has found many applications in several fields. Biometric data matching [1, 2], recommender systems [3, 4], data mining [5, 6], and data aggregation [7, 8] are only a few examples. Instead of implicitly assuming that all processing parties are trusted, SPED provides an environment for the parties, e.g. server and client, to collaborate for processing privacy-sensitive information in a privacy-preserving manner. The main idea in SPED is to provide only the encrypted version of the data to the server, and invoke interactive cryptographic protocols between the server and the decryption key owner to perform the desired algorithms.

Although SPED protects the privacy of sensitive information, its computational and communication overhead are the main challenges in large scale applications. Actually, the SPED algorithms are attractive from privacy preservation perspective but they are far more complex than their plaintext versions since the core operations like comparison, equality tests and division, which are repeated in large quantities, are computationally demanding. Therefore, reducing the complexity of such operations is an important challenge to make the cryptographic solutions practical. As a result, a number of protocols have been introduced [9–12].

In this work, we address secure equality testing (EQT), one of the fundamental operations needed in many SPED solutions, e.g. for search algorithms over encrypted data [13], which is addressed previously in [11, 14, 15]. In our scenario, Alice holds two encrypted values while Bob has the decryption key. Our aim is to design a cryptographic protocol for Alice and Bob that outputs a single encrypted bit, the result of the equality test, which is also secret for both. We propose two EQT protocols based on [11, 15]. We introduce algorithmic changes and data packing that improve the performance of the existing work significantly as shown in the complexity analysis. Experimental results also show that our protocols outperform the existing work, and present up to 99% runtime improvement in a fair experimental setup.

The rest of the paper is organized as follows. We introduce the notation, the cryptographic building blocks, and the security assumptions in Section 2.2. In Section 1.3, we describe the related work. We describe our proposals, two EQT protocols, in Section 1.4, and provide complexity and security analyses in Section 1.5. We demonstrate the performance of the proposals in terms of computational complexity and run-time in Section 1.6. Finally, we conclude in Section 2.6.

1.2. PRELIMINARIES

In this section, we describe the application setting, the security assumptions, and the cryptographic tools used in this article. We summarize our notation in Table III.1.1.

Table II.1.1: Symbols and their meaning.

Symbol	Description	Symbol	Description
a, b	input messages	ρ	number of plaintext can packed into a ciphertext
x_i	the ith bit of integer x		
ℓ	bit length of inputs	n	crypto system modulus
κ	statistical security parameter	$\binom{i}{e}$	binomial coefficients
$r, \grave{r}, R$	random numbers	ϑ	result of the equality test
pk	public key	sk	private key
$d, \grave{d}$	Hamming distance	$[.]$	Paillier encryption
α_i	coefficients of Lagrange polynomial	$\oplus$	exclusive-OR
N	number of equality tests	$u_{(\lambda)}$	u exponentiations with λ-bit exponents

1.2.1. HOMOMORPHIC ENCRYPTION

In this article, we rely on an additively homomorphic cryptosystem, more specifically the Paillier cryptosystem [16]. An additively homomorphic encryption scheme preserves certain structure that can be exploited to process ciphertexts without decryption. Given $\mathscr{E}_{pk}(m_1)$ and $\mathscr{E}_{pk}(m_2)$, a new ciphertext whose decryption yields the sum of the plaintext messages m_1 and m_2 can be obtained by performing a multiplication operation over the ciphertexts under additively homomorphic encryption schemes: $\mathscr{D}_{sk}(\mathscr{E}_{pk}(m_1) \otimes \mathscr{E}_{pk}(m_2)) = m_1 + m_2$.

Consequently, exponentiation of any ciphertext with a public value yields the encrypted product of the original plaintext and the exponent: $\mathscr{D}_{sk}(\mathscr{E}_{pk}(m)^e) = e \cdot m$.

In the rest of the paper, we denote the ciphertext of a message m by $[m]$ for the Paillier cryptosystem.

1.2.2. SECURITY ASSUMPTIONS AND THE SETTING

We consider the semi-honest security model, where parties are honest in following the protocol steps, while they can keep messages to deduce more information than they are entitled to. We assume that Bob has the decryption key sk and Alice has the encryptions of two values a and b, which are ℓ-bit integers each. The values a and b are secret and should be kept hidden from Alice and Bob. However, Alice should obtain an encrypted bit $\vartheta \leftarrow (a = b)?1 : 0$, where ϑ is the result of the equality test which is also kept secret from Alice and Bob. During the computation of ϑ, the intermediate values should also be kept secret from both parties to limit the information leakage.

1.3. RELATED WORK

The ideas behind existing equality testing include using Hamming distance, quadratic residuosity assumption, and bit-decomposition. Takashi and Kazuo [14] proposed a probabilistic constant-round equality test protocol, where Jacobi symbol is used to test quadratic residuosity of a value. Schoenmakers and Tuyls [11] have shown a practical method to check the equality of two encrypted values by introducing a bit-decomposition protocol. In [15], Lipmaa and Toft have introduced an equality test protocol, which is

based on computing Hamming distance between two private values. The protocol uses Lagrange interpolation [17] to generate a polynomial, which is used to obtain the result of the equality test, and a multiplicative masking method described in [18]. In the following, we present [15] and [11] that inspired us to design our protocols, which are also deterministic unlike [14].

1.3.1. EQT BASED ON THE HAMMING DISTANCE (LT13)

The equality testing protocol in [15] is based on computing the Hamming distance of two private values. LT13 computes a polynomial in order to obtain an encrypted bit, which represents the result of the equality testing protocol as given in Protocol 1.

Protocol 1 LT13

1: Alice computes $[a-b]$, additively masks the result with a random number r, and sends $[x] \leftarrow [a-b+r]$ to Bob.
2: Bob decrypts the message $[x]$ and computes the first ℓ bits x_i, encrypts them separately, and sends them to Alice.
3: Alice computes the Hamming distance $[d]$ between $[x_i]$ and $[r_i]$, masks it, and returns the masked encrypted value $[y]$ to Bob. The Hamming distance is zero if and only if $a = b$, since in this case $x = r$. Note that d is masked multiplicatively with the inverse of a random number r.
4: Bob decrypts $[y]$ and computes the exponentiations y^i, $1 < i \leq \ell$. The y^i are encrypted and sent back to Alice.
5: Alice unmasks the $[y^i]$ to obtain $[d^i]$ and computes ℓ-degree Lagrange polynomial $[\vartheta] \leftarrow [\sum_{i=0}^{\ell} \alpha_i \cdot d^i]$, where α_i are coefficients that depend on ℓ.

1.3.2. EQT BASED ON THE BIT-DECOMPOSITION (ST06)

The protocol in [11] decomposes $[z] \leftarrow [a-b]$ to its encrypted bits $[z_i]$, $0 \leq i < \ell$, and then computes $[\prod_{i=0}^{\ell-1}(1-z_i)]$, which is [1] in case of equality, by using a secure multiplication protocol as summarized in Protocol 2.

Protocol 2 ST06

1: Alice computes $[z] \leftarrow [a-b]$, masks the result additively, and sends $[x] \leftarrow [z+r]$ to Bob.
2: Bob decrypts $[x]$, computes the first ℓ bits, encrypts them separately, and sends $[x_i]$, $0 \leq i < \ell$, to Alice.
3: Alice runs a secure subroutine, based on a secure multiplication protocol, to compute the first ℓ bits of $[z]$ (as explained in Protocol 6), namely $[z_i]$, $0 \leq i < \ell$.
4: Alice runs another subroutine to compute $[\vartheta] \leftarrow [\prod_{i=0}^{\ell-1}(1-z_i)]$.

1.4. IMPROVED SECURE EQUALITY TESTS

We now describe two equality testing protocols based on LT13 and ST06. We achieve this by introducing a novel secure exponentiation protocol and employing data packing.

1.4.1. IMPROVED EQT BASED ON THE HAMMING DISTANCE (NEL-I)

To improve LT13 in terms of computational complexity, first, we introduce a novel method of computing exponentiation securely, then we use data packing [19] to decrease the decryption cost [10]. Note that packing technique can be used when we have multiple equality tests to perform at once. Furthermore, we add one more round to the protocol, which results in a significant decrease in computational complexity. In fact, the degree of the polynomial that Alice has to compute decreases from ℓ to $\lceil \log_2 \ell \rceil$.

In LT13, Alice computes the Hamming distance d between x and r, and then generates the polynomial based on d, which results in generating an ℓ-degree polynomial. Computing an ℓ-degree polynomial over encrypted data introduces significant computational overhead. To decrease the degree of the polynomial, we add one more round in NEL-I : As shown in Protocol 3, Alice computes the Hamming distance $[d]$, then she masks it with a random value $[y] \leftarrow [d + \grave{r}]$. Afterwards, she computes the Hamming distance $[\grave{d}]$ between $[y]$ and $[\grave{r}]$. Since $0 \leq \grave{d} \leq \lceil \log_2 \ell \rceil$, Alice constructs a $\lceil \log_2 \ell \rceil$-degree polynomial using the specified mapping and computes the values $[\grave{d}^i]$, which are used later to obtain $[\vartheta]$.

Another computationally demanding part of LT13 is the secure exponentiation method, which is required in order to compute $[\vartheta] \leftarrow [\sum_{i=0}^{c} \alpha_i \cdot d^i]$. In the secure exponentiation protocol, Alice has $[d]$ and she has to compute $[d^i]$ with Bob's help, who has the private key. Although the secure exponentiation method in LT13, given in Protocol 4, is simple and straightforward, unmasking, $[d^i] \leftarrow [t^i]^{R^i}$, is very expensive. In fact, the unmasking dominates the overall computational complexity of LT13. The unmasking in Protocol 3 is an extended form of a method used in [2], which is also computationally expensive. In this work, we introduce a novel secure exponentiation method, Protocol 5, which makes the unmasking significantly less expensive. As it is shown in Protocol 5 (which replaces Step 7 in Protocol 3), the additive masking is used instead of multiplicative form to blind $[d]$. Then, we use an efficient method to unmask the encrypted values and obtain $[d^i]$. Note that unlike the secure exponentiation in LT13, where $[d^\ell]$ can be obtained directly, we need to compute all $[d^i]$, $1 < i < \ell$, before computing $[d^\ell]$ in NEL-I. However, both LT13 and NEL-I demand computation of $[\sum_{i=0}^{\ell} \alpha_i \cdot d^i]$, which requires computing all $[d^i]$, $1 < i \leq \ell$.

1.4.2. IMPROVED EQT BASED ON THE BIT-DECOMPOSITION (NEL-II)

Ignoring the decryption cost, ST06 has a very low computational complexity compared to LT13. However, the number of times that Bob invokes decryption in ST06 is very high, which makes the protocol computationally expensive. To improve the performance, we propose a variant of ST06 that is also based on the bit-decomposition method but employs data packing, introducing a significant improvement in computation. We provide the details in Protocol 6.

1.5. SECURITY ANALYSIS

In this section, we argue that our algorithmic changes and deploying packing technique do not violate the security of the protocols in the semi-honest security model as long as the underlying cryptographic primitive is secure. Furthermore, In order to show that

Protocol 3 NEL-I

1: Alice generates a sufficiently large ($\ell+\kappa$ bits) random value r, computes $[x] \leftarrow [a-b+r]$ and sends $[x]$ to Bob. Alice puts multiple x values into a single ciphertext by using data packing. Assume that (a,b) and r are ℓ and $\ell+\kappa$ bits integers, respectively. Then, $[x]$ is a $(\ell+\kappa+1)$-bit integer. Let the message space of the Paillier cryptosystem be n, then Alice packs $\rho = \lfloor n/(\ell+\kappa+1)\rfloor$ into one Paillier message as follows:

$$[\hat{x}] = \sum_{j=0}^{\rho-1} [x_j \cdot (2^{\ell+\kappa+1})^j], \tag{1.1}$$

and then sends $[\hat{x}]$ to Bob.

2: Bob decrypts $[\hat{x}]$ and unpacks it. Then, he computes the first ℓ bits x_i, for $0 \le i < \ell$, of each component; encrypts them separately and sends $[x_i]$ to Alice.

3: Alice computes $[r_i \oplus x_i]$ for $0 \le i < \ell$ and then $[d] \leftarrow [\sum_{i=0}^{\ell-1} r_i \oplus x_i] = \prod_{i=0}^{\ell-1}[r_i \oplus x_i]$, which is the Hamming distance of r and x (note that $[r_i \oplus x_i] \leftarrow [x][r][x]^{-2r}$). Then, she additively masks $[d]$ with an $(\lceil \log_2 \ell\rceil + \kappa)$-bit random number $\grave{r}$ that is $[y] \leftarrow [d+\grave{r}]$ and sends $[y]$ to Bob in packed form $[\hat{y}]$.

4: Bob decrypts $[\hat{y}]$ and unpacks it. Then, he computes the first $\lceil \log_2 \ell\rceil$ bits y_i, $0 \le i < \lceil \log_2 \ell\rceil$ of each component, encrypts them separately, and sends the $[y_i]$ to Alice.

5: Alice computes $[\grave{r}_i \oplus y_i]$ for $0 \le i < \lceil \log_2 \ell\rceil$ and then $[\grave{d}] \leftarrow [\sum_{i=0}^{\lceil \log_2 \ell\rceil-1} \grave{r}_i \oplus y_i] = \prod_{i=0}^{\lceil \log_2 \ell\rceil-1}[\grave{r}_i \oplus y_i]$. Then, she additively masks $[\grave{d}]$ with another $(\lceil \log_2 \log_2 \ell\rceil + \kappa)$-bit random number R, computes $[t] \leftarrow [\grave{d}+R]$, and sends $[t]$ to Bob in packed form $[\hat{t}]$.

6: Bob decrypts $[t]$, computes t^i, $1 < i \le \lceil \log_2 \ell\rceil$, and sends t^i to Alice in encrypted form.

7: Alice unmasks $[t^i]$ by computing $[\grave{d}^i] \leftarrow [t^i - \sum_{e=1}^{i}\binom{i}{e}\grave{d}^{i-e}R^e] = [t^i][\prod_{e=1}^{i}[\grave{d}^{i-e}]^{\binom{i}{e}R^e}]$, $1 < i \le \lceil \log_2 \ell\rceil$.

8: Alice constructs a $\lceil \log_2 \ell\rceil$-degree polynomial $[\vartheta] \leftarrow [\sum_{i=0}^{\lceil \log_2 \ell\rceil} \alpha_i \cdot \grave{d}^i] = \prod_{i=0}^{\lceil \log_2 \ell\rceil}[\grave{d}^i]^{\alpha_i}$, where it maps $\grave{d}=1$ to 1, and $\grave{d} \in \{2,3,\ldots,\lceil \log_2 \ell\rceil\}$ to 0.

the protocols are privacy-preserving, we need to show Alice and Bob cannot learn new private information from each other. Recall that Alice only receives encrypted messages from Bob and she cannot distinguish ciphertexts, since the encryption scheme used in the protocols is semantically secure. Thus, it suffices to show that there is no information leakage to Bob in order to prove the improved equality testing protocols are privacy-preserving. It is clear that using packing technique does not leak any information since it uses homomorphic properties of the Paillier crypto-scheme. Therefore, we need to show that additive blindings used in the NEL-I are secure. In Steps 1, 3, and 5 of Protocol 3, Alice blinds her encrypted values additively before sending them to Bob. Thus, the decrypted messages in Bob are statistically indistinguishable from the original values before blinding. For blinding an ℓ-bit value a additively, Alice chooses a random value r that is κ bits longer than the actual a ($\kappa = 80$ bits), and then computes $a+r$. The security proof of additive blinding is related to statistical indistinguishability of $x = a+r$ from a random number x_R, which is drawn uniformly from $\{0,1,\ldots,2^{\ell+\kappa+1}\}$, as described in [9].

Protocol 4 Secure exponentiation (LT13)

Input: $[d]$
Output: $[d^i]$ for $1 < i \leq \ell$

1: Alice chooses a random number $R \in \mathbb{Z}_N^*$, computes its inverse R^{-1}, and R^i, $1 < i \leq \ell$. Then, Alice multiplicatively masks $[d] \leftarrow [d+1]$ with R^{-1} that is $[t] \leftarrow [d+1]^{R^{-1}}$ and sends $[t]$ to party B. Note that 1 is added to $[d]$ to make sure $d \in \mathbb{Z}_{N^2}^*$, which is needed to get the correct result after performing unmasking.
2: Party B decrypts $[t]$, computes t^i, $1 < i \leq \ell$, and sends t^i to party A in encrypted form.
3: Party A unmasks $[t^i]$ by computing $[d^i] \leftarrow [t^i]^{R^i}$ for $1 < i \leq \ell$.

Protocol 5 Secure exponentiation (NEL-I)

Input: $[d]$
Output: $[d^i]$ for $1 < i \leq \ell$

1: Alice chooses a random number R, where R is a $(\lceil \log_2 d \rceil + \kappa)$-bit value, and then she sends $[t] \leftarrow [d+R]$ to Bob.
2: Party B decrypts $[t]$, computes t^i, $1 < i \leq \ell$, and sends t^i to party A in encrypted form.
3: Alice has $[d]$ and R values and she can easily compute $[p_1] \leftarrow [dR]$, and then $[d^2] \leftarrow [t^2 - 2p_1 - R^2]$. To compute $[d^3]$, she computes $[p_1] \leftarrow [d^2 R]$ and $[p_2] \leftarrow [p_1 R]$, and then $[d^3] \leftarrow [t^3 - \sum_{e=1}^{2} \binom{3}{e} p_e - R^3]$. In order to obtain $[d^\ell]$, she computes $[p_1] \leftarrow [d^{\ell-1} R]$ and $[p_i] \leftarrow [p_{i-1} R]$, $1 < i < \ell$, and then she computes $[d^\ell] \leftarrow [t^\ell - \sum_{e=1}^{\ell-1} \binom{\ell}{e} p_e - R^\ell]$.

1.6. PERFORMANCE ANALYSIS

In this section, we compare the performance of the EQT protocols based on analyzing computational complexities and the experimental results.

1.6.1. COMPUTATIONAL COMPLEXITY

Tables IV.1.3 and II.1.3 present the computational complexities of the secure equality testing protocols in terms of multiplication and exponentiation. As an example, the exponentiation complexity of the ST06 protocol is $(6\ell)_{-1}$, which means there are 6ℓ exponentiations with a negative 1-bit exponents. It is clear that LT13 is the most expensive protocol because of the unmasking technique described in 4. To simplify the complexities and compare the protocols easier, we represent the complexities of exponentiation as multiplication. We can represent a ciphertext modulo n with an x-bit exponent as $3x/2$ multiplications modulo n. In Table II.1.3, overall complexity shows the complexity of each protocol represented as the number of multiplications. It can be observed that LT13 has a polynomial complexity, while ST06, NEL-I, and NEL-II are linear. Note that the complexities of encryption and decryption are not included in Tables IV.1.3 and II.1.3 since the protocols are crypto-scheme-independent and homomorphic crypto-schemes may have different encryption and decryption complexities.

Protocol 6 NEL-II

1: Alice computes $[z] \leftarrow [a-b]$, $[x] \leftarrow [z+r]$, where r is a $(\ell+1+\kappa)$-bit random number, and sends $[x]$ to Bob in the packed form.
2: Bob decrypts and unpacks $[x]$, decomposes first ℓ bits x_i, $0 \le i < \ell$, and sends $[x_i]$ to Alice.
3: Alice computes $[c_0] \leftarrow [x_0]^{r_0}$, $[z_0] \leftarrow [x_0][r_0][c_0]^{-2}$, and sets $i = 1$.
4: Alice chooses $\ell-2$ random bits R to mask $[c_{i-1}]$ such that $[\theta] \leftarrow [c_{i-1} \oplus R]$. Afterward, Alice sends $[\theta]$ to Bob. Note that the $[\theta]$ are encrypted one-bit values, which means Alice can pack n messages into one ciphertext, $\rho = n$ that decreases the Paillier decryption and communication costs significantly.

$$[\hat{\theta}] = \sum_{j=0}^{n-1} [\theta_j \cdot 2^j], \tag{1.2}$$

5: Bob decrypts and unpacks the $[\hat{\theta}]$ to obtain θ, then computes $\alpha \leftarrow \theta \times x_i$ and sends $[\alpha]$ to Alice.
6: Alice unmasks $[\alpha]$ to obtain $[\beta_i] \leftarrow [c_{i-1}] \otimes [x_i]$ by distinguishing $R = 0$ and $R = 1$. If $R = 0$, $[\beta_i] \leftarrow [\alpha]$, else $[\beta_i] \leftarrow [1][\alpha]^{-1}$. Then, Alice computes $[c_i] \leftarrow [x_i]^{r_i}[c_{i-1}]^{r_i}[\beta_i]^{1-2r_i}$ and $[z_i] \leftarrow [x_i][r_i][c_{i-1}][c_i]^{-2}$. Afterward, Alice sets $i \leftarrow i+1$ and jumps to step 4 until $i = \ell-1$. In order to get the equality testing result Alice computes $[\vartheta] \leftarrow \prod_{i=0}^{\ell-1}[1-z_i]$ by running secure multiplication protocol as follows:
7: Alice chooses two random bits r_i and r_j and computes $[\alpha] \leftarrow [z_i \oplus r_i]$ and $[\beta] \leftarrow [z_j \oplus r_j]$. Then, Alice sends $[\alpha]$ and $[\beta]$ to Bob in packed form. Similar to step 4, using packing technique decreases number of decryption and communication cost significantly.
8: Bob decrypts and unpacks $[\alpha]$ and $[\beta]$, multiply them, and sends $[\theta] \leftarrow [\alpha \times \beta]$ to Alice.
9: Alice computes $[z_i \times z_j] \leftarrow [\theta][z_i]^{2r_ir_j-r_j}[z_j]^{2r_ir_j-r_i}[-r_ir_j]^{1/(1-2r_i-2r_j+4r_ir_j)}$.

Table IV.1.3 and II.1.3 also presents the complexities of the secure exponentiation protocols used in LT13 and NEL-I, that are LT13(Expo) and NEL-I(Expo), respectively. Clearly, our new unmasking technique reduces the complexity of LT13(Expo) from $\mathcal{O}(n^\ell)$ to $\mathcal{O}(\ell^2 \log \ell)$. Note that the d value used in NEL-I(Expo) is between 0 and ℓ, resulting the exponential complexity to be $(\ell(\ell-1)/2)_{(\lceil \log_2 \ell/2 \rceil+\kappa)} + (2(\ell-1))_{-1}$. However, in NEL-I, the input of the secure exponentiation protocol is $\grave{d}$ that is between 0 and $\lceil \log_2 \ell \rceil$. Given the range of $\grave{d}$ in NEL-I, the complexity of the secure exponentiation protocol is $(\lceil \log_2 \ell \rceil(\lceil \log_2 \ell \rceil - 1)/2)_{(\lceil \log_2(\log_2 \ell/2) \rceil+\kappa)} + (2(\lceil \log_2 \ell \rceil - 1))_{-1}$, which is $\mathcal{O}((\log \ell)^2)$.

Table II.1.4 presents the number of decryptions of the protocols, where ST06 has the highest number of decryptions. Table II.1.4 shows that NEL-II has far fewer number of decryption compared to ST06, which results in a significantly much more efficient protocol when we consider the decryption cost.

Table II.1.2: Computational complexities of the secure equality testing and the exponentiation protocols.

Protocols	Multiplication	Exponentiation
LT13	$(3/2)\ell+3$	$1_{-1}+(\ell/2)_{-1}+1_{(n/2)}+\sum_{i=2}^{\ell-1}1_{(n/2)^i}$ $+\ell_{(\ell/2)}$
ST06	$11\ell+\ell/2-1$	$(6\ell)_{-1}$
NEL-I	$1/2(\lceil\log_2\ell\rceil(\lceil\log_2\ell\rceil+4)+\ell)$ $+3\rho$	$3\rho_{\ell+\kappa}+(\lceil\log_2\ell\rceil(\lceil\log_2\ell\rceil-1)/2)_{\kappa}$ $+((\ell+\lceil\log_2\ell\rceil)/2)_{-1}$
NEL-II	$11\ell+\ell/2+3\rho-1$	$(6\ell)_{-1}+\rho_{\ell+\kappa}+(2\rho)_2$
LT13(Expo)	1	$1_{(n/2)}+\sum_{i=2}^{\ell-1}1_{(n/2)^i}$
NEL-I(Expo)	$(\ell^2+3\ell)/2-2$	$(\ell(\ell-1)/2)_{(\lceil\log_2\ell/2\rceil+\kappa)}$ $+(2(\ell-1))_{-1}$

Table II.1.3: Overall complexity of the secure equality testing and the exponentiation protocols.

Protocols	Overall complexity	
LT13	$(1024)^{\ell-1}+3\ell^2/4+3\ell+1540$	$\mathcal{O}(n^{\ell})$
ST06	$71\ell/2-1$	$\mathcal{O}(\ell)$
NEL-I	$\lceil\log_2\ell\rceil(120\lceil\log_2\ell\rceil-115)+7\ell/2+9366$	$\mathcal{O}(\ell)$
NEL-II	$71\ell/2+3321$	$\mathcal{O}(\ell)$
LT13(Expo)	$(1042)^{\ell-1}+1537$	$\mathcal{O}(n^{\ell})$
NEL-I(Expo)	$3/4\ell^2\lceil\log_2\ell/2\rceil+60\ell^2-55\ell-5$	$\mathcal{O}(\ell^2\log\ell)$

1.6.2. EXPERIMENTAL RESULTS

We implemented the protocols using C++ and external libraries: MPIR, Boost, and SeComLib on a single Linux machine running Ubuntu 14.04 LTS, with 64-bit microprocessor and 8 GB of RAM. The cryptographic key length of the Paillier is chosen according to NIST standards [20], which are valid until 2030. Table III.1.3 shows the parameters used in the implementation of the secure equality testing protocols. We analyze the performance of the protocols with different input sizes.

1.6.3. COMPUTATION OF α_i IN NEL-I

In this section, we show that α_i values in Protocol 3 (step 8) can be precomputed efficiently. Note that this section is not included in the original paper.

In the step 8, Alice constructs a $\lceil\log_2\ell\rceil$-degree polynomial $[\vartheta]\leftarrow[\sum_{i=0}^{\lceil\log_2\ell\rceil}\alpha_i\cdot\grave{d}^i]$, where

Table II.1.4: Decryption complexities of the protocols.

Protocols	Decryption
LT13	$2N$
ST06	$N(3\ell-1)$
NEL-I	$N(\ell+\kappa+1)/n$
NEL-II	$\ell+N(\ell+\kappa+4)/n+2\lceil\log_2\ell\rceil$

Table II.1.5: Parameters used in the implementation.

Parameter	Symbol	Value
Bit size of inputs	ℓ	2-30
Number of performed equality test	N	1000
Security parameter	κ	80 bits
Paillier message space	n	2048 bits

it maps $\grave{d} = 1$ to 1, and $\grave{d} \in \{2, 3, \ldots, \lceil \log_2 \ell \rceil\}$ to 0. To construct the polynomial, we used Lagrange polynomial. Generating a polynomial that outputs the pairs $(x_0, y_0), \ldots, (x_k, y_k)$ by using Lagrange polynomial is as follows:

$$F(x) = \sum_{i=0}^{k} (y_i \cdot L_i(x))$$

$$L_i(x) = \prod_{0 \le j \le k \,\&\, j \ne i} \frac{x - x_j}{x_i - x_j}$$

In general, the generated polynomial has rational coefficients. Computation of a polynomial with rational coefficients in encrypted domain results in expensive cryptographic operations. However, we are interested in a polynomial that generates $(1, 1)$, $(2, 0), \ldots, (x_k, 0)$. Thus, the generated polynomial will be:

$$F(x) = \sum_{i=0}^{k} (y_i \cdot L_i(x)) = y_0 \cdot L_0(x),$$

$$F(x) = \prod_{0 \le j \le k \,\&\, j \ne 0} \frac{x - x_j}{1 - x_j} = \frac{\prod_{z=2}^{k} (x - z)}{(-1)^{k-1}(k-1)!}$$

Then, we can simply compute the inverse of $(-1)^{k-1}(k-1)! \bmod n$, where $k = \lceil \log_2 \ell \rceil$. Moreover, computation of $\prod_{z=2}^{k} (x - z)$ results in a polynomial with integer coefficients. Therefore, α_i values can be precomputed efficiently.

REFERENCES

[1] A. Sadeghi, T. Schneider, and I. Wehrenberg, *Efficient privacy-preserving face recognition*, in *Information, Security and Cryptology - ICISC 2009, 12th International Conference, Seoul, Korea, December 2-4, 2009, Revised Selected Papers* (2009) pp. 229–244.

[2] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, *Privacy-preserving face recognition*, in *Privacy Enhancing Technologies, 9th International Symposium, PETS 2009, Seattle, WA, USA, August 5-7, 2009. Proceedings* (2009) pp. 235–253.

[3] R. L. Lagendijk, Z. Erkin, and M. Barni, *Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation*, IEEE Signal Process. Mag. **30**, 82 (2013).

[4] A. Jeckmans, A. Peter, and P. Hartel, *Efficient privacy-enhanced familiarity-based recommender system,* in *Computer Security–ESORICS 2013* (Springer, 2013) pp. 400–417.

[5] B. Pinkas, *Cryptographic techniques for privacy-preserving data mining,* SIGKDD Explorations **4**, 12 (2002).

[6] Z. Yang, S. Zhong, and R. N. Wright, *Privacy-preserving classification of customer data without loss of accuracy,* in *Proceedings of the 2005 SIAM International Conference on Data Mining, SDM 2005, Newport Beach, CA, USA, April 21-23, 2005* (2005) pp. 92–102.

[7] C. Li, R. Lu, H. Li, L. Chen, and J. Chen, *Pda: a privacy-preserving dual-functional aggregation scheme for smart grid communications,* Security and Communication Networks **8**, 2494 (2015).

[8] Z. Erkin and G. Tsudik, *Private computation of spatial and temporal power consumption with smart meters,* in *Applied Cryptography and Network Security - 10th International Conference, ACNS 2012, Singapore, June 26-29, 2012. Proceedings* (2012) pp. 561–577.

[9] T. Veugen, *Encrypted integer division,* in *2010 IEEE International Workshop on Information Forensics and Security, WIFS 2010, Seattle, WA, USA, December 12-15, 2010* (2010) pp. 1–6.

[10] M. Nateghizad, Z. Erkin, and R. L. Lagendijk, *An efficient privacy-preserving comparison protocol in smart metering systems,* EURASIP J. Information Security **2016**, 11 (2016).

[11] B. Schoenmakers and P. Tuyls, *Efficient binary conversion for paillier encrypted values,* in *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings* (2006) pp. 522–537.

[12] G. Couteau, *Efficient secure comparison protocols,* Cryptology ePrint Archive, Report 2016/544 (2016), `http://eprint.iacr.org/2016/544`.

[13] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, *Secure ranked keyword search over encrypted cloud data,* in *2010 International Conference on Distributed Computing Systems, ICDCS 2010, Genova, Italy, June 21-25, 2010* (2010) pp. 253–262.

[14] T. Nishide and K. Ohta, *Multiparty computation for interval, equality, and comparison without bit-decomposition protocol,* in *Public Key Cryptography–PKC 2007* (Springer, 2007) pp. 343–360.

[15] 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.

[16] 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.

[17] I. Damgård, M. Fitzi, E. Kiltz, J. B. Nielsen, and T. Toft, *Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation,* in *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings* (2006) pp. 285–304.

[18] J. Bar-Ilan and D. Beaver, *Non-cryptographic fault-tolerant computing in constant number of rounds of interaction,* in *Proceedings of the Eighth Annual ACM Symposium on Principles of Distributed Computing, Edmonton, Alberta, Canada, August 14-16, 1989* (1989) pp. 201–209.

[19] 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).

[20] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, *Nist sp800-57: Recommendation for key management part 1: General (revised),* NIST, Tech. Rep (2007).

II.2 | Privacy-Preserving Equality Testing Protocols

Abstract

Protocols for securely testing the equality of two encrypted integers are common building blocks for a number of proposals in the literature that aim for privacy preservation. Being used repeatedly in many cryptographic protocols, designing efficient equality testing protocols is important in terms of computation and communication overhead. In this work, we consider a scenario with two parties where party A has two integers encrypted using an additively homomorphic scheme and party B has the decryption key. Party A would like to obtain an encrypted bit that shows whether the integers are equal or not but nothing more. We propose three secure equality testing protocols, which are more efficient in terms of communication, computation or both compared to the existing work. To support our claims, we present experimental results, which show that our protocols achieve up to 99% computation-wise improvement compared to the state-of-the-art protocols in a fair experimental set-up.

This chapter has been published as "Secure Equality Testing Protocols in the Two-Party Setting", by M.Nateghizad, T.Veugen, Z.Erkin and R.L.Lagendijk in the proceedings of 13th *International Conference on Availability, Reliability and Security*, 2018. This work was nominated for the best paper award.

2.1. INTRODUCTION

Processing encrypted data has been addressed in several fields, e.g. biometric data matching [1], recommender systems [2], data mining [3] and data aggregation [4], as it enables collaborating with an untrustworthy service provider to process privacy-sensitive data. The main idea is to provide only the encrypted version of the data to the service provider and invoke interactive cryptographic protocols with the decryption key owner to perform the desired algorithm. While being very secure regarding protecting the privacy-sensitive data without hampering the service, processing encrypted data introduces a significant amount of computational and communication overhead compared to performing the same algorithm with unencrypted data. In the literature, it is suggested to design custom-tailored cryptographic protocols, rather than applying generic solutions, to improve the efficiency of the privacy-preserving version of the algorithm. Building blocks with encrypted data for those algorithms like comparison, division, and equality checks [5] need to be designed with high efficiency. The main reason is that these core operations are repeated in large quantities in conventional data processing algorithms. For example, finding similar users in a system with millions of users to a particular one based on his or her taste for movies [2] requires comparison of similarity scores linear in the number of users in the system. Testing the equality of two encrypted integers is one of the widely-used operations, e.g. for searching in encrypted databases. Other applications for equality testing protocols also include, but are not limited to, secure pattern matching [6], secure linear algebra [7], and encryption switching protocols [8].

Yang et al. [9] introduced the first public key encryption that supports testing equality (PKwET). Their work allows checking whether two ciphertexts encrypted under the same or different keys are encryptions of the same value. Tong[10] introduced a protocol to enable equality testing for authorized users. Later works tried to improve the performance or functionality of PKwET [11–13]. However, existing PKwET proposals leak the result of the equality test to the service provider, which is usually a semi-trusted remote computation server.

Secure multi-party computation (MPC) is another approach to design algorithms for secure equality checking. In such protocols, two or more parties jointly compute an agreed function of their secret inputs. Many works have been introduced to show that any function can be computed securely using MPC [14–16]. Nishide and Ohta proposed a probabilistic constant-round equality test protocol [17], where the Jacobi symbol is used to test quadratic residuosity of a value. Although the proposed protocol, (NO07), is efficient regarding computation, the result of the protocol is probabilistic: with a probability of 50%, the protocol returns a correct answer. The protocol is suggested to be executed φ times to minimize the error probability to $2^{-\varphi}$. As it is necessary to pick large values for φ to reduce the false positive rate, the protocol becomes computation and communication wise demanding.

Schoenmakers and Tuyls [18] has presented a method (ST06) to check the equality of two encrypted integers by using a protocol based on bit-decomposition. However, this protocol is expensive regarding the number of communication rounds. In [5], Lipmaa and Toft have introduced an equality test protocol (LT13) on top of an arithmetic black box [19]. The protocol uses Lagrange interpolation similar to [20]. A multiplicative masking is used in [20], a similar idea from [21], since its realization is easier than

additive masking. However, multiplicative masking is not computationally efficient in a two-party setting because the size of the exponent can be very large. However, it is important to note that LT13 is more efficient when there are more than two parties in the setting. The works [18] and [5] are both secure against active adversaries.

There are also efficient secure equality testing protocols based on Garbled circuits (GC) [22, 23]. In [22], Kolesnikov et al. propose an efficient construction that enables XOR to be evaluated for free. Then, in [23], evaluation of MPC protocols (VAT09) using free XOR technique has improved the computational efficiency of garbled circuits up to 50%. However, computational efficiency was achieved at the cost of significant communication and pre-computation overhead. Moreover, using GC results in additional computation and communication overhead for converting encryptions to garbled circuit inputs as explained in [23].

In this paper, our aim is to check whether two encrypted values under an additively homomorphic encryption scheme such as Paillier [24] are equal or not. More precisely, party A has the encryption of two ℓ-bit integers, $[a]$ and $[b]$, where $[\cdot]$ denotes the encryption, and party B has the decryption key. Neither party is allowed to learn the outcome bit ϑ. The bit ϑ is one, exactly when $a = b$, and zero otherwise, similar to the other existing works [5, 17, 18]. We assume $0 \leq a, b < 2^{\ell}$.

We propose three secure equality testing protocols, namely EQT-1, EQT-2 and EQT-3. As it is shown in Figure II.2.1, there is a trade-off between computation and communication cost. For the sake of clarity, we present a summary of our protocols:

- **EQT-1** is based on a coin toss where the results determines either performing a secure Hamming distance computation for the two inputs or invoking a secure joint function [25]. The resulting protocol has the least communication overhead among all other proposals. While it is computationally more expensive than our other two proposal, it is computationally 97% more efficient than the state-of-the-art and requires 24% less communication overhead.

- **EQT-2** relies on computing the Hamming distance and the secure comparison protocol [26]. The protocol has a balanced computational and communication overhead. EQT-2 is computationally 38% more efficient than EQT-1 with 25% more communication overhead.

- **EQT-3** is using Lagrange interpolation that has the best overall computational performance and 99% more efficient than the state-of-the-art. EQT-3 is computation-wise over 30% more efficient than EQT-2, but it has 43% more communication overhead compared to EQT-2.

2.2. PRELIMINARIES

The symbols and their description are listed in Table IV.1.2.

2.2.1. SECURITY SETTING

We consider the semi-honest security model [27], where both parties are assumed to be honest in following the protocol description, while they are curious to obtain more

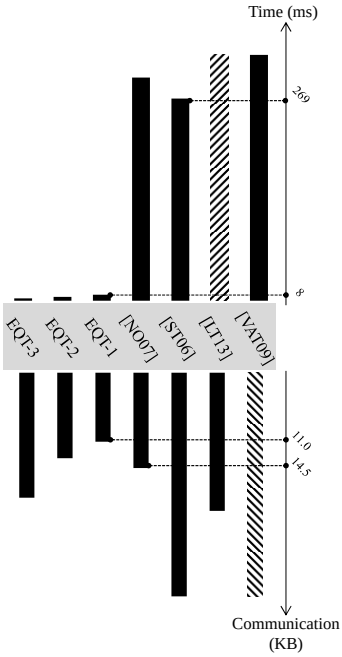

Figure II.2.1: A summary of secure equality testing protocols' performance for 20-bit inputs. Dashed bar denotes that the value is very large and does not fit in the graph.

information than they are entitled to. In this setting, it is assumed that A and B do not collude.

2.2.2. HOMOMORPHIC ENCRYPTION

We use two additively homomorphic and semantically secure encryption schemes, namely Paillier [28] and DGK [25]. In an additively homomorphic encryption scheme, multiplying two ciphertexts $\mathscr{E}_{pk}(m_1)$ and $\mathscr{E}_{pk}(m_2)$ results in a ciphertext, whose decryption is the sum of two plaintexts m_1 and m_2: $\mathscr{D}_{sk}(\mathscr{E}_{pk}(m_1) \cdot \mathscr{E}_{pk}(m_2)) = (m_1 + m_2) \bmod n$, where n is the encryption system modulus. Consequently, exponentiation of any ciphertext with a public integer value k yields the encrypted product of the original plaintext and the public value: $\mathscr{D}_{sk}(\mathscr{E}_{pk}(m)^k) = (k \cdot m) \bmod n$.

PAILLIER

The Paillier encryption [28] for a given message $m \in \mathbb{Z}_n$ is defined as $\mathscr{E}_{pk}(m, r) = g^m \cdot r^n \bmod n^2$, where n is the product of two distinct large prime numbers p and q, ciphertext $\mathscr{E}_{pk}(m, r) \in \mathbb{Z}^*_{n^2}$, r is a random number from $\mathbb{Z}^*_n$, and g is a generator of $\mathbb{Z}^*_n$. The public key is (g, n), and the private key is (p, q). This encryption scheme is additively homomorphic. The security of Paillier is based on hardness of computing n^{th} residue classes. For the decryption operation, we refer readers to [28].

Table II.2.1: List of symbols

Symbol	Description	Symbol	Description
a, b	input messages	δ	comparison result
ℓ	input bit size	n	crypto modulus
$\oplus$	exclusive or	κ	security parameter
r, ρ, s, w	randoms	φ	error controller [17]
$[.]$	Paillier cipher	$[\![.]\!]$	DGK cipher
A, B	stack holders	x_i	i^{th} bit of integer x
$\mathbb{Z}_u$	DGK plaintext space	$d(a, b)$	Hamming distance of a and b
p, q, v_p, v_q	primes	t	DGK parameter
ϑ	equality result	sk	private key
pk	public key	$\mathscr{E}_{pk}$	encryption
$\mathscr{D}_{sk}$	decryption	$\log \ell$	$\lceil \log_2 \ell \rceil$

DGK

The DGK cryptosystem [25] is used in this work for two reasons: 1) it is more efficiency than Paillier in term of computation and communication since it has much smaller ciphertext size, 2) it enables checking whether a ciphertext is an encryption of zero without performing the expensive decryption operation, which can save computation. Note that DGK decryption is very expensive for large inputs since it uses a look-up table.

The process of generating the keys is as follows: 1) choose two distinct t-bit prime numbers v_p, v_q, 2) construct two distinct prime numbers p and q, where $v_p|(p-1)$ and $v_q|(q-1)$ such that $n = pq$ is a k-bit RSA modulus, 3) choose u, the smallest possible prime number but greater than $\ell+2$, 4) choose a random r that is a $2.5t$-bit integer, and 5) choose g and h such that $\text{ord}(g) = uv_pv_q$ and $\text{ord}(h) = v_pv_q$, where $\ell < t < k$. The public and the private keys are $pk = (n, g, h, u)$ and $sk = (p, q, v_p, v_q)$, respectively.

In the rest of the paper, we denote the ciphertext of a message m by $[m]$ for the Paillier cryptosystem and $[\![m]\!]$ for the DGK. We also omit the modular reductions, when describing the computational steps, for simplicity.

2.3. OUR PROTOCOLS

2.3.1. EQUALITY TESTING PROTOCOL (EQT)-1

In this protocol, described in Protocol 14, we use the idea that computes either the Hamming distance between two encryptions or performs a secure comparison based on the idea from the DGK comparison protocol [25], after a coin toss. The reason for this coin toss is as follows: calculating the Hamming distance is less expensive regarding computation and communication overhead. However, only using Hamming distance for equality check leaks information. Assume that only the Hamming distance is used for testing the equality of a and b. Then, party B learns whether $a = b$ after performing the DGK zero-check, since the Hamming distance is always zero, precisely when $a = b$. Party B acquiring this information is not desired since we do not want Party A and B to learn any information. Therefore, we toss a coin and we either compute the Hamming distance or

perform secure comparison, hiding what is being computed from Party B.

Protocol 7 EQT-1

1: Party A generates a sufficiently large ($\ell + 1 + \kappa$ bits) random value r, computes $[x] \leftarrow [a - b + r]$, and sends $[x]$ to B.
2: Party B decrypts $[x]$, computes the first ℓ bits x_i, $0 \le i < \ell$, encrypts them separately with DGK (for efficiency reason), and sends $[\![x_i]\!]$ to A.
3: Party A computes $[\![r_i \oplus x_i]\!]$ for $0 \le i < \ell$, by distinguishing $r_i = 0$ and $r_i = 1$. If $r_i = 0$, $[\![r_i \oplus x_i]\!] \leftarrow [\![x_i]\!]$, else, $[\![r_i \oplus x_i]\!] \leftarrow [\![1]\!] \cdot [\![x_i]\!]^{-1}$.
4: Party A tosses a random coin $\delta_A \in \{0, 1\}$.
5: **if** $\delta_A = 0$ **then**
6: Party A computes $[\![c_0]\!] \leftarrow \prod_{i=0}^{\ell-1} [\![r_i \oplus x_i]\!]$, and multiplicatively blinds it with a large (in case of DGK, the random number ρ should contain $2t$ bits [29]) random number ρ.
7: Party A generates $\ell - 1$ non-zero random integers c_i, $1 \le i < \ell$, and encrypts them.
8: **else**
9: Party A computes $[\![c_i]\!] \leftarrow [\![-1]\!] \cdot [\![r_i \oplus x_i]\!] \cdot (\prod_{j=i+1}^{\ell-1} [\![r_j \oplus x_j]\!])^2$, for $0 \le i < \ell$.
10: Party A generates ℓ large (in case of DGK, the random numbers ρ_i should be $2t$ bits) non-zero random numbers ρ_i, $0 \le i < \ell$, and uses them to multiplicatively blind c_i: $[\![c_i]\!] \leftarrow [\![c_i]\!]^{\rho_i}$.
11: **end if**
12: Party A randomly permutes the order of the $[\![c_i]\!]$, $0 \le i < \ell$, and sends them to party B.
13: Party B decrypts (in case of DGK, a DGK zero-check is sufficient) the numbers $[\![c_i]\!]$ to find whether one of them is 0. If (at least) one of them is 0, party B sets $\delta_B \leftarrow 1$, otherwise $\delta_B \leftarrow 0$.
14: Party B encrypts δ_B, and sends it to A.
15: Party A computes $[\vartheta]$, by distinguishing $\delta_A = 0$ and $\delta_A = 1$. If $\delta_A = 0$, $[\vartheta] \leftarrow [\delta_B]$, else, $[\vartheta] \leftarrow [1] \cdot [\delta_B]^{-1}$.

CORRECTNESS

Since $0 \le a, b < 2^\ell$, we have $r - 2^\ell < x = a - b + r < r + 2^\ell$, so it is sufficient to check whether the first (least significant) ℓ bits of x and r are equal. We know $\sum_{i=0}^{\ell-1} r_i \oplus x_i \ge 0$, with equality precisely when $a = b$. In case $\delta_A = 0$, we have $c_0 = \sum_{i=0}^{\ell-1} r_i \oplus x_i$, so the (blinded) numbers c_i, $0 \le i < \ell$ will have exactly one zero, precisely when $a = b$, and will all be non-zero, otherwise. Therefore, if $\delta_B = 1$, we have $\delta_A \oplus \delta_B = 1 = (a = b)$. The case $\delta_B = 0$ is similar. If $\delta_A = 1$, party A computes $c_i = -1 + r_i \oplus x_i + 2\sum_{j=i+1}^{\ell-1} r_j \oplus x_j$. If all $r_i \oplus x_i = 0$, then all $c_i = -1$, and $\delta_B = 0$. Otherwise, precisely one of the c_i will be zero [30], and $\delta_B = 1$. Therefore, in both cases $(a = b) \leftarrow \delta_A \oplus \delta_B$.

OPTIMIZATION

Although the decryption of $[x]$ and the multiplicative blindings dominate the computational complexity, the number of multiplications for computing the $[\![c_i]\!]$ can be reduced further.

Instead of computing $c_i \leftarrow -1 + r_i \oplus x_i + 2\sum_{j=i+1}^{\ell-1} r_j \oplus x_j$, we can use the optimization introduced in [26]. Parties can compute $c_i \leftarrow -1 + r_i + x_i + \sum_{j=i+1}^{\ell-1} 2^j (r_j - x_j) = (-1 + r_i + \sum_{j=i+1}^{\ell-1} 2^j r_j) + (x_i - \sum_{j=i+1}^{\ell-1} 2^j x_j) = c_i^A + c_i^B$, where c_i^A can be computed in clear by party A, and c_i^B by party B. Then, for each i, $0 \le i < \ell$, it requires only one multiplication to compute $[\![c_i]\!] \leftarrow [\![c_i^A]\!] \cdot [\![c_i^B]\!]$, which party A has to perform in case $\delta_A = 1$. These modified c_i's retain the property that precisely one of them is zero, if and only if, $a \neq b$.

In case $\delta_A = 0$, party A can use the same $[\![c_0^B]\!]$ to compute an encryption of $c_0 \leftarrow (-r_0 + \sum_{j=1}^{\ell-1} 2^j r_j) + c_0^B$ instead of $c_0 \leftarrow \sum_{i=0}^{\ell-1} r_i \oplus x_i$, which requires computing $c_0^A \leftarrow -r_0 + \sum_{j=1}^{\ell-1} 2^j r_j$ in clear. Also for this c_0, we have $c_0 = 0$, if and only if $a = b$. This optimization avoids computing $[\![r_i \oplus x_i]\!]$ by party A, and reduces the computation of each $[\![c_i]\!]$ to one multiplication.

2.3.2. EQUALITY TESTING PROTOCOL (EQT)-2

Our proposal also uses the Hamming distance of a and b, which is $d(a,b) \leftarrow \sum_{i=0}^{\ell-1} a_i \oplus b_i$, to determine ϑ. As described in Protocol 8, it works as follows: (1) party A computes $[x]$, (2) party A computes the encrypted Hamming distance between $x \bmod 2^\ell$ and $r \bmod 2^\ell$, and (3) party A and B use a secure comparison protocol to compute the encryption of $(d > 0)$. If $d > 0$, then $a \neq b$, and $a = b$, otherwise. Note that there is a fundamental difference with EQT-1, which also computes Hamming distance: EQT-2 computes the Hamming distance of the two encrypted inputs, invokes a secure comparison protocol, namely EPPCP from [26], which returns an encrypted result. Therefore, there is no information revealed to Party B. The choice of using EPPCP is based on its high performance. The DGK comparison protocol is not preferred as the message space too large to create a look-up table.

Protocol 8 EQT-2

1: Party A generates a sufficiently large ($\kappa + \ell + 1$ bits) random value r, computes $[x] \leftarrow [a - b + r]$, and sends $[x]$ to B.
2: Party B decrypts $[x]$, computes the first ℓ bits x_i, $0 \le i < \ell$, and their sum $X \leftarrow \sum_{i=0}^{\ell-1} x_i$, encrypts them all separately, and sends them to A.
3: Party A computes $[d] \leftarrow [\sum_{i=0}^{\ell-1} r_i \oplus x_i] = [\sum_{i=0}^{\ell-1} r_i] \cdot [X] \cdot (\prod_{i=0, r_i=1}^{\ell-1} [x_i])^{-2}$.
4: Two parties jointly run the comparison protocol (EPPCP) [26], where party A receives $[1]$ if $[0 < d]$, $[0]$ otherwise, while both parties learn nothing about the inputs and the relation between $[d]$ and $[0]$. The ones' complement of the result of EPPCP is simply the result of the equality test, $[\vartheta] \leftarrow [1] \cdot (EPPCP([0],[d]))^{-1}$.

CORRECTNESS

Since $x - r = a - b$, and $0 \le a, b < 2^\ell$, then $a = b$ if $d = d(x \bmod 2^\ell, r \bmod 2^\ell) = 0$. To check if $a = b$, party A and party B jointly run EPPCP, where $EPPCP$ returns (encrypted) zero when $a = b$, and (encrypted) one when $a \neq b$.

OPTIMIZATION

In EQT-2, we first securely compute $d \leftarrow d(a,b)$, $0 \le d \le \ell$, and afterward securely compute $\delta \leftarrow (d > 0)$. The complexity of EPPCP depends on the size of its inputs. To reduce this size, we can add an additional communication round to securely compute $\acute{d} \leftarrow d(d,0)$, $0 \le \acute{d} < \log_2 \ell$, and securely compare $\acute{d}$ with 0. This way of reducing the input size can be repeated many times, which reduces the computation effort at the cost of increasing the number of communication rounds.

2.3.3. EQUALITY TESTING PROTOCOL (EQT)-3

Previously presented protocols rely on secure comparison or efficient zero-check of the DGK encryption scheme. Unlike the other two protocols, EQT-3, described in Protocol 9, is a protocol that does not require zero-checking or secure comparison. The main idea of EQT-3 is to first compute the Hamming distance e between $x \bmod 2^\ell$ and $r \bmod 2^\ell$ similar to the other protocols. Note that $0 \le e \le \ell$, where ℓ is the bit-length of the inputs a and b. To make the range of e smaller (later we show that the smaller range of e results in a significant more efficient protocol), we mask $[e]$ with a large number number w, $[y] \leftarrow [e+w]$. Afterward, we compute the Hamming distance d between $y \bmod 2^{\log_2 \ell}$ and $w \bmod 2^{\log_2 \ell}$, where $0 \le d \le \log_2 \ell$. Finally, we generate and compute a polynomial that maps $d = 0$ to 1, and $d \in \{1, 2, \ldots, \log_2 \ell\}$ to 0.

Protocol 9 EQT-3

1: Party A generates a sufficiently large ($\ell + 1 + \kappa$ bits) random value r, computes $[x] = [a - b + r]$, and sends $[x]$ to B.
2: Party B decrypts $[x]$, computes the first ℓ bits x_i, $0 \le i < \ell$, and their sum $X = \sum_{i=0}^{\ell-1} x_i$, encrypts them separately, and sends them to A.
3: Party A computes the first ℓ bits of r to derive $[e] = [\sum_{i=0}^{\ell-1} r_i \oplus x_i] = [\sum_{i=0}^{\ell-1} r_i] \cdot [X] \cdot (\prod_{i=0, r_i=1}^{\ell-1} [x_i])^{-2}$.
4: Party A generates a sufficiently large ($\log_2 \ell + \kappa$ bits) random value w, computes $[y = e + w]$, and sends $[y]$ to B.
5: Party B decrypts $[y]$, computes the first $\log_2 \ell$ bits y_i, $0 \le i < \log_2 \ell$, and their sum $Y = \sum_{i=0}^{(\log_2 \ell)-1} y_i$, encrypts them separately, and sends them to A.
6: Party A computes the first $\log_2 \ell$ bits of w to derive $[d] = [\sum_{i=0}^{(\log_2 \ell)-1} w_i \oplus y_i] = [\sum_{i=0}^{(\log_2 \ell)-1} w_i] \cdot [Y] \cdot (\prod_{i=0, w_i=1}^{(\log_2 \ell)-1} [y_i])^{-2}$.
7: Party A generates a sufficiently large ($\log_2 \log_2 \ell + \kappa$ bits) random value s, computes $[z = d + s]$, and sends $[z]$ to B.
8: Party B decrypts $[z]$, computes $\lambda = z \bmod \varrho$, where $\varrho = (\log_2 \ell) + 1$, and from that the integers γ_i, $0 \le i < (2\log_2 \ell) + 1$, as specified bellow. Party B encrypts the γ_i, and sends them to A.
9: Party A computes $\sigma = s \bmod \varrho$, and $[\vartheta] = [f(\sigma - \lambda)] = [\sum_{i=0}^{2\log_2 \ell} \gamma_i \sigma^i] = (\ldots([\gamma_{2\log_2 \ell}]^\sigma \cdot [\gamma_{(2\log_2 \ell)-1}])^\sigma \ldots \cdot [\gamma_1])^\sigma \cdot [\gamma_0]$.

The polynomial f, which is specified below, can be computed beforehand in the clear. Each integer γ_i can be computed by only one multiplicative inverse and one multiplication. The exponentiation to the power -2 in step 3 is implemented by one multi-

plicative inverse, and one square.

COMPUTATION OF γ_i

As shown before, $x = a - b + r$, $e = d(x, r)$, $y = e + w$, $d = d(y, w)$, $z = d + s$, and finally $\vartheta = (z = s)$, where $0 \le d \le \log \ell$. The idea is to compute the Lagrange polynomial $f(x)$ such that it maps 0 to 1, and maps x to 0, where $0 < |x| \le \log_2 \ell$. Then, we can compute $\delta = f(s \bmod \varrho - z \bmod \varrho) = f(\sigma - \lambda)$.

The Lagrange polynomial f is easily found as $f(x) = \prod_{i=-\log_2 \ell, i\neq 0}^{\log \ell} \frac{x-i}{-i} = (-1)^{\log \ell} (\log \ell!)^{-2} \prod_{i=-\log \ell, i\neq 0}^{\log \ell} (x - i) = (-1)^{\log \ell} (\log \ell!)^{-2} \sum_{i=0}^{2\log \ell} f_i x^i$, where $f_i \in \mathbb{Z}$ can be derived.

The binomial expansion of $x^i = (\sigma - \lambda)^i$, for $0 \le i \le 2\log \ell$, gives $\sum_{j=0}^{i} \binom{i}{j} \sigma^j (-\lambda)^{i-j}$. Therefore, we can write

$$f(\sigma - \lambda) = (-1)^{\log \ell} (\log \ell!)^{-2} \sum_{i=0}^{2\log \ell} f_i \sum_{j=0}^{i} \binom{i}{j} \sigma^j (-\lambda)^{i-j},$$

which reduces to $\sum_{j=0}^{2\log \ell} \gamma_j \sigma^j$, where $\gamma_j = (-1)^{\log \ell} (\log \ell!)^{-2} \sum_{i=j}^{2\log \ell} f_i \binom{i}{j} (-\lambda)^{i-j}$ and $0 \le j \le 2\log \ell$. In order to compute the γ_j, party B computes the multiplicative inverse of $(\log \ell!)^2$ modulo n, and multiplies this with the integer $\sum_{i=j}^{2\log \ell} f_i \binom{i}{j} (-\lambda)^{i-j}$.

CORRECTNESS

In EQT-3, the Hamming distance e between $x \bmod 2^\ell$ and $r \bmod 2^\ell$ is computed. For efficiency purpose (we will discuss this later), then, the second Hamming distance d between the first $\log \ell$ low significant bits of $e + w$ and w is computed. Afterward, party B computes encrypted γ_i values, which are required from party A to compute a polynomial that outputs the equality testing result. Actually, a Lagrange polynomial $f(\sigma - \lambda) = \prod_{i=-\log \ell, i\neq 0}^{\log \ell} \frac{(\sigma-\lambda)-i}{-i}$ is generated in this equality testing protocol, where it maps ($\sigma - \lambda = 0$) to 1 and other values to 0. Recall that $\sigma = s \bmod \varrho$, $\lambda = z \bmod \varrho$, and $z = d + s$; therefore, if and only if $d = 0$, then $f(\sigma - \lambda)$ outputs 1.

OPTIMISATION

Similar to ETQ-3, adding one more round reduces the communicational and computational costs. Besides that, it is also possible to lessen the number of rounds in EQT-3 if there is a limit in a particular application setting. However, decreasing one round makes the polynomial more complicated and the protocol more expensive.

2.4. SECURITY ANALYSIS

In this section, we provide proofs to show that our three secure equality testing protocols are simulation secure in the semi-honest security model. Informally, we mean that the probability that an adversary can learn private information from truly generated data by the parties in our protocols is at most negligibly more than the probability that an adversary can learn from given randomly generated data. We use the simulatability paradigm [31] in our proofs, where the adversary takes the control of the network and try to obtain

the final result of the protocol by itself as the only party in the protocol. In this paradigm, security is defined as a comparison of computation work-flow in “real world" and “ideal world".

In real world, a protocol can be broken into sub-protocols or computations that are carried out by each party throughout the protocol. Let us denote π as one of EQT protocols; we can split π into two parts: $\pi = \{\pi_A, \pi_B\}$, which are performed in parties A and B, respectively. π_A takes $[a]$ and $[b]$, which are the inputs, and outputs $[\vartheta]$, $[\vartheta] \leftarrow \pi_A([a],[b];\phi)$. And π_B decrypts the given encryptions from party A, processes them, and sends their encrypted versions to party B. Thus, to perform secure equality testing the encrypted messages flow from one party to another party and together they generate the $[\vartheta]$ as the result of EQT. Assuming party A is corrupted by an adversary $\mathscr{A}$, then $\mathscr{A}$ has access to $[a]$, $[b]$, and $[\vartheta]$. Similarly, when party B is corrupted, the adversary has access to the intermediate computation results.

In an ideal world, it is assumed that one of the parties is corrupted by an adversary. Then, he uses a simulator to generate the outputs of the other party. This would be similar to performing EQT with just one party, which is corrupted. In the ideal world, an adversary $\acute{\mathscr{A}}$, who has control over party A, has only access to her inputs $[a]$, $[b]$, and the garbage inputs given from simulated party B instead of the correct result of π_B. The goal is to show that $\mathscr{A}$ can learn equal or negligibly more than $\acute{\mathscr{A}}$, meaning that they are computationally indistinguishable, then we can claim that EQT is a simulation secure protocol.

Definition 2.4.1. Let $a \in \{0,1\}^*$ represents the parties' inputs, $n \in \mathbb{N}$ to be a security parameter, and $X = \{X(a,n)\}_{a\in\{0,1\}^*;n\in\mathbb{N}}$ and $Y = \{Y(a,n)\}_{a\in\{0,1\}^*;n\in\mathbb{N}}$, two infinite sequences of random variables, are probability ensembles. Then, X and Y are computationally indistinguishable, denoted as $X \overset{c}{\equiv} Y$, if there is a polynomial $p(.)$ for every non-uniform polynomial-time probabilistic algorithm (nuPPT) D such that:

$$|Pr[D(X(a,n)) = 1] - Pr[D(Y(a,n)) = 1]| < 1/p(n) \tag{2.1}$$

2.4.1. SECURITY OF EQT-1

Let denote the computation of $[c_i]$ as A_{f_1}, δ_B as B_{f_1}, and $[\vartheta]$ as A_{f_2} in EQT-1. Let $f = (A_f, B_f)$, where $A_f = (A_{f_1}, A_{f_2})$ and $B_f = (B_{f_1})$, the f to be the PPT functionality for EQT-1. The view of the ith party, $i \in \{A,B\}$, during the execution of EQT-1 on $([a],[b];\phi)$ and security parameter n is denoted by $view_i^{EQT-1}([a],[b];\phi;n) = (w, r_i; m_1^i, \cdots, m_t^i)$, where $w \in \{[a],[b],\phi\}$ based on the value of i, r^i are the ith party internal random numbers, and m_j^i represents the jth message that is received by i^{th} party. Recall that party B does not have any initial input, thus its inputs is denoted as ϕ. $output_i^{EQT-1}([a],[b];\phi;n)$ represents the output of each party in EQT-1. To represent the joint output of both parties, we denote

$$\begin{aligned} output^{EQT-1} = (&output_1^{EQT-1}([a],[b];\phi;n), \\ &output_2^{EQT-1}([a],[b];\phi;n)). \end{aligned} \tag{2.2}$$

Definition 2.4.2. EQT-1 securely computes $f = (A_f, B_f)$ in the semi-honest security set-

ting if there exits PPT algorithms Sim_A and Sim_B such that:

$$\{(Sim_A(1^n,[a],[b],A_f,f))\} \overset{c}{\equiv} \{(view_A^f ([a],[b];\phi;n), output^f([a],[b];\phi;n))\} \tag{2.3}$$

and

$$\{(Sim_B(1^n,\phi,B_f,f))\} \overset{c}{\equiv} \{(view_B^f ([a],[b];\phi;n), output^f([a],[b];\phi;n))\} \tag{2.4}$$

Theorem 1. *The protocol EQT-1 is simulation secure and securely computes the functionality f, when the party A is corrupted by adversary $\mathscr{A}$ in the presence of semi-honest adversaries.*

We need to show that party A cannot computationally distinguish between generated messages and outputs from $\mathscr{S}_2$ that is the simulation of party B, and randomly generated data. Party A receives an output from $\mathscr{S}_2$, $[\delta_B]$. Given $[a]$, $[b]$, and 1^n (security parameter tape), party A works as follow:

1. Party A chooses uniformly distributed random number r, δ_A, and r_i, $i \in \{0,\cdots,\ell-1\}$ for A_f.
2. Party A executes A_{f_1} to obtain $[c_i]$, and sends them to S_2.
3. The simulator of party B, S_2, tosses a random coin $\acute{\delta}_B \in \{0,1\}$ and sends $[\acute{\delta}_B]$ to party A.
4. Party A performs A_{f_2} based on given $[\acute{\delta}_B]$ to get $[\acute{\vartheta}]$.

The output of the simulation can be written as: $Sim_A(1^n,[a],[b],A_f$ $,f) = ([a],[b],r,\delta_A,r_i;[\acute{\delta}_B];([\acute{\vartheta}],\phi))$. The real view of part A can be presented as $view_A^f([a],[b]) = ([a],[b],r,\delta_A,r_i;[\delta_B])$. And the output of the real view is $output^f([a],[b]) = ([\vartheta],\phi)$. It can be observed that the encryption pairs $([\acute{\delta}_B],[\acute{\vartheta}])$ and $(\delta_B,[\vartheta])$ are computationally indistinguishable, since the crypto-scheme used in EQT-1 is semantically secure. Therefore, we can claim that

$$Sim_A(1^n,[a],[b],A_f,f) \overset{c}{\equiv} \{view_A^f([a],[b];\phi), output^f([a],[b];\phi)\}. \tag{2.5}$$

Theorem 2. *The protocol EQT-1 is simulation secure and securely computes the functionality f, when the party B is corrupted by adversary $\mathscr{A}$ in the presence of semi-honest adversaries.*

1. S_1 chooses ℓ uniformly distributed random integers $\acute{c}_i$ and encrypts them, $[\acute{c}_i]$.
2. S_1 tosses a random coin $r_1 \in \{0,1\}$. If $r_1 = 1$, then S_1 chooses a uniformly distributed random number $r_2 \in \{0,\cdots,\ell-1\}$ and sets $[\acute{c}_{r_2}] \leftarrow [0]$.
3. S_1 sends $[\acute{c}_i]$ to party B.

4. Party B executes B_{f_1} and sends $[\delta_B]$ back to S_1.

The simulation and the real view can be written as:

$$\begin{aligned} Sim_B(1^n, \phi, B_{f_1}, f) &= (\phi; [\acute{c}_i]; ([\vartheta], \phi)) \\ view_B^f([a], [b], \phi, n) &= (\phi; [c_i]; ([\vartheta].\phi)) \end{aligned} \tag{2.6}$$

Since party A has the decryption key, we should show that $\mathscr{A}$ cannot distinguish between $\acute{c}_i$ and c_i. To do so, we analyze two possible types of information leakage:

1. Existence of zero in c_i: in EQT-1, $\mathscr{A}$ cannot learn extra information if, for any i, $c_i = 0$. The reason is that δ_A decides whether $c_i = 0$ means the equality or inequality of a and b, and party B has no access to δ_A. Moreover, location i does not leak any information, because party A permutes the c_i values before sending them to party B.

2. Information about a and b if $c_i \neq 0$: in case of $c_i \neq 0$, party B is still cannot learn any extra information, since the c_i values are multiplicatively masked in party A.

Therefore, adversary $\mathscr{A}$ cannot computationally distinguish between c_i and $\acute{c}_i$, which means:

$$Sim_B(1^n, \phi, B_{f_1}, f) \overset{c}{\equiv} \{view_B^f([a], [b], \phi, n), output^f([a], [b]; \phi)\} \tag{2.7}$$

2.4.2. SECURITY OF EQT-2

Denoting computation of $[x]$ as A_{f_1}, $([X], [x_i])$ as B_{f_1}, $[g]$ as A_{f_2}, $([\hat{g}], [t_i], [g.2^{-\log_2 \ell}])$ as B_{f_2}, $[e_i]$ as A_{f_3}, $[\hat{\lambda}]$ as B_{f_3}, and $[\vartheta]$ as A_{f_4}, we have $A_f = (A_{f_1}, A_{f_2}, A_{f_3}, A_{f_4})$, $B_f = (B_{f_1}, B_{f_2}, B_{f_3})$, and $f = (A_f, B_f)$.

Theorem 3. *The protocol EQT-2 is simulation secure and securely computes the functionality f, when the party A is corrupted by adversary $\mathscr{A}$ in the presence of semi-honest adversaries.*

1. Party A chooses uniformly random numbers r, $\hat{r}$, s, and h_i.

2. Party A executes A_{f_1} to obtain $[x]$ and sends it to S_2

3. S_2 generate ℓ random one-bit values $\hat{x}$ and another random integer $\hat{X}$. S_2 sends $[\hat{x}_i]$ and $[\hat{X}]$ to party A.

4. Party A calls A_{f_2} to get $[g]$ and sends it to S_2.

5. S_2 generate three random numbers $[\acute{g}]$, $[\hat{t}_i]$, and $[g^{''}]$ to party A.

6. Party A executes A_{f_3}, computes $[e_i]$, and sends $[e_i]$ to S_2.

7. S_2 tosses a random coin $\hat{\lambda}$ and sends it to party A.

8. Party A executes A_{f_4} to obtain $[\vartheta]$.

Based on the same reason in Theorem 1, clearly, party A cannot distinguish between $([\hat{x}],[\hat{X}],[\acute{g}],[\hat{t}_i],[g''],[\acute{\lambda}])$ and $([x],[X],[g],[t_i],[g\cdot 2^{-\log_2 \ell}],[\hat{\lambda}])$. Therefore,

$$Sim_A(1^n,[a],[b],A_f,f) \overset{c}{\equiv} \{view_A^f([a],[b];\phi), output^f([a],[b];\phi)\}. \tag{2.8}$$

Theorem 4. *The protocol EQT-2 is simulation secure and securely computes the functionality f, when the party B is corrupted by adversary $\mathscr{A}$ in the presence of semi-honest adversaries.*

Party A receives three outputs from $\mathscr{S}_1$ $[x]$, $[g]$, and $[e_i]$. Given $[1]$, $[b]$, and 1^n (security parameter), party A works as follow:

1. S_1 chooses a $(\kappa+\ell+1)$-bit random value $\hat{x}$, a random value $o \in \{-\ell,\cdots,\ell\}$ and sends $[\hat{x}] \leftarrow [\hat{x}+o]$ it to party B.

2. Party B executes F_{f_1} and sends $[x_i]$ and $[X]$ back to S_1.

3. S_1 generates a $(\kappa+\log_2 \ell+1)$-bit random value $\acute{g}$ and sends it to party B.

4. Party B call F_{f_2} and sends $[\hat{g}]$, $[t_i]$, and $[g\cdot 2^{-\log_2 \ell}]$ to S_1.

5. S_1 chooses i random number $\hat{e}_i \in \mathbb{Z}_u^*$, $i \in \{0,\cdots,\log_2 \ell-1\}$. Then, S_1 tosses another coin δ, and if $\delta=1$, then chooses a random i and $\hat{e}_i=0$. Afterwards, S_1 sends $[\hat{e}_i]$ to party B.

6. Party B executes B_{f_3} to get $\acute{\lambda}$, and sends it to S_1.

The simulation and the real view can be written as:

$$\begin{aligned} sim_B(1^n,\phi,B_f,f) &= (\phi;[\hat{x}],[\acute{g}],[\hat{e}_i];([\vartheta,\phi])) \\ view_B^f([a],[b],\phi,n) &= (\phi,[x],[g],[e_i];([\vartheta,\phi])) \end{aligned} \tag{2.9}$$

To provide simulation security, party B should not be able to distinguish between $(\hat{x},\acute{g},\hat{e}_i)$ and (x,g,e_i). Recall that party B has access to the decryption key and can see the data in the clear. 1) party A masks $[a-b]$ with a large enough random value r to hide the difference from party B. Thus, $\mathscr{A}$ cannot distinguish between x and $\hat{x}$. 2) $[z]$ values are also additively masked with another random number, which makes $\acute{g}$ and g indistinguishable for party B. 3) party A also multiplicatively masks c_i values which also makes $\hat{e}$ computationally indistinguishable from e to party B [25, 32]. 4) party B cannot learn about the relation between a and b be seeing a zero in one of the e_i values, since it is calculated based on random number s. Thus, if any of $e_i=0$ then still party B does not know whether $a=b$ or $a\neq b$. Based on the four stated properties, we can claim that:

$$Sim_B(1^n,\phi,B_f,f) \overset{c}{\equiv} \{view_B^f([a],[b],\phi,n), output^f([a],[b];\phi)\} \tag{2.10}$$

2.4.3. SECURITY OF EQT-3

Let us denote computation of $[x]$ as A_{f_1}, $[x_i]$ and $[X]$ as B_{f_1}, $[y]$ as A_{f_2}, $[y_i]$ and $[Y]$ as B_{f_2}, computation of $[z]$ as A_{f_3}, $[\gamma]$ as B_{f_3}, and computation of $[\vartheta]$ as A_{f_4}.

Theorem 5. *The protocol EQT-3 is simulation secure and securely computes the functionality f, when the party A is corrupted by adversary $\mathscr{A}$ in the presence of semi-honest adversaries.*

S_2 works as follow:

1. Party A generate uniformly distributed random numbers r, w and s.
2. Party A executes A_{f_1} to obtain $[x]$ and sends it to S_2.
3. S_2 generate uniformly distributed random numbers $[\hat{x}_i]$ and $[\hat{X}]$, and sends them to party A.
4. Party A calls A_{f_2} and sends $[\hat{y}]$ to S_2.
5. S_2 generate random number $[\hat{y}_i]$ and $[\hat{Y}]$, and sends them to party A.
6. Party A performs A_{f_3} to get $[z]$ and sends it S_2.
7. S_2 chooses a random number $\hat{\gamma}$ and sends $[\hat{\gamma}]$ to party A.
8. Party A executes A_{f_4} to obtain $[\vartheta]$.

Because of semantical security of the crypto-schemes used in this work, $\mathscr{A}$ cannot distinguish between $([\hat{x}_i],[\hat{X}],[\hat{y}_i],[\hat{Y}],[\hat{\gamma}])$ and $([x_i],[X],[y_i],[Y],[\gamma])$. Thus, we can claim that

$$Sim_A(1^n,[a],[b],A_f,f) \stackrel{c}{\equiv} \{view_A^f([a],[b];\phi), output^f([a],[b];\phi)\}. \tag{2.11}$$

Theorem 6. *The protocol EQT-3 is simulation secure and securely computes the functionality f, when the party A is corrupted by adversary $\mathscr{B}$ in the presence of semi-honest adversaries.*

S_1 works as follow:

1. S_1 chooses a random number $[\hat{x}]$.
2. Party B receives $[\hat{x}]$, decrypts it, executes B_{f_1} to obtain x_i and X, and sends them to S_1 in encryption form.
3. S_1 generates random number $\hat{y}$ and sends $[\hat{y}]$ to party B.
4. Party B executes B_{f_2} and sends $[y_i]$ and $[Y]$ back to S_1.
5. S_1 chooses a random number $\hat{z}$ and sends $[\hat{z}]$ to party B.
6. Party B calls B_{f_3} to get γ and sends it to S_1 in encryption form.

Recall that party B is keeping the private key and is able to decrypts the encryption given from party A. Thus, we need to show $\mathscr{A}$ cannot learn any private information by distinguishing given messages. First, party B cannot distinguish between $\hat{x}$ and x, since x is additively masked with a $\kappa+\ell+1$-bit value. Second, distinguishing between $\hat{y}$ and y is not computationally possible for party B, since y is additively masked with large enough random value. Similarly, party B cannot distinguish between $\hat{z}$ and z, since z value is additively masked. Therefore, we can conclude that

$$Sim_B(1^n, \phi, B_f, f) \stackrel{c}{\equiv} \{view_B^f([a],[b],\phi,n), output^f([a],[b];\phi)\} \tag{2.12}$$

2.5. PERFORMANCE ANALYSIS

2.5.1. COMPLEXITY ANALYSIS

Communication. Table II.2.2 presents the number of communication rounds and the amount of data transmitted. Table II.2.2 shows that, except ST06 and NO07, they are all constant-round protocols. ST06 uses bit decomposition, which results in more number of communication rounds. According to Table II.2.2, using EQT-1 requires the least

Table II.2.2: Number of communication rounds and amount of data transmission ($\ell = 20, \varphi = 12, u = 31$).

Protocols	Rounds		Transmitted data (KB)	
[LT13]	2	2	$\ell+1$	21
[ST06]	$2\ell+2\lceil\log_2\ell\rceil+2$	52	$(6\ell-5)/2$	57.5
[NO07]	3φ	36	$(2\varphi+\lceil\log_2\varphi\rceil+1)/2$	14.5
EQT-1	2	2	$(\ell+2)/2$	1 1
EQT-2	3	3	$(\ell+\lceil\log_2\ell\rceil)/2+2$	14.5
EQT-3	3	3	$(\ell+3\lceil\log_2\ell\rceil+6)/2$	20

data transmission among the other protocols, which is mainly due to the use of DGK encryption scheme. In contrast, EQT-3 is the least communication-wise efficient protocol because of the large ciphertext space of Paillier and transmission of the Lagrange polynomial coefficients.

COMPUTATION.

Table II.2.3 presents the overall computational cost given in the number of modular multiplications. The cost of the DGK zero-check function can be represented as $3t/4$ multiplications modulo n [25] and we can show the complexity of a ciphertext modulo n with an x-bit exponent as $3x/2$ multiplications modulo n. According to Table II.2.3, LT13 is the most computation-wise expensive protocol because of having ℓ exponentiations with $(2\kappa)i$ bits exponents, for $1 \le i \le \ell$. There are also $(6\ell)_{-1}$ exponentiations in ST06, but the exponents are -1. From the Table II.2.3, we can see that EQT-3 has the least number of multiplication among the others. The main reason for having this outstanding efficiency in EQT-3 is performing less expensive exponentiations. In EQT-3, there are $(2\lceil\log_2\ell\rceil)$ exponentiations with $\lceil\log_2\ell\rceil/2$-bit exponents.

Our proposed protocols show different performances regarding communication and computation. For a system with limited communication resources, EQT-1 is a suitable

Table II.2.3: Overall computational cost and the complexity ($\ell = 20$).

Protocols	Multiplication		Complexity
[LT13]	$\ell(768(\ell+1)+7/2)+3079$	331169	$\mathcal{O}(\ell^2)$
[ST06]	$45\ell+\ell/2-1$	902	$\mathcal{O}(\ell)$
[NO07]	$(3\ell\lceil\log_2 \ell!\rceil)/2+\ell+20672$	22522	$\mathcal{O}(\ell\log_2 \ell!)$
EQT-1	$504\ell+\ell/2+345$	10435	$\mathcal{O}(\ell)$
EQT-2	$\ell/2+841\lceil\log_2 \ell\rceil+25$	4240	$\mathcal{O}(\ell)$
EQT-3	$\ell/2+3/2(\lceil\log_2 \ell\rceil)^2+5/2\lceil\log_2 \ell\rceil+14$	74	$\mathcal{O}(\ell)$

choice, since it has only two communication rounds and the lowest data transmission cost. Although EQT-1 has a very low communication cost, its computational cost is twice more than EQT-2 and hundred times more than EQT-3. For a system with very limited computational resources, EQT-3 is a good choice, since it has significantly low computational cost. However, EQT-3's data transmission cost is twice more than EQT-1 and also higher than EQT-2.

2.5.2. EXPERIMENTAL RESULTS

The protocols are implemented using C++ and external libraries: MPIR, Boost, and SeComLib on a single Linux machine running Ubuntu 14.04 LTS, with a 64-bit microprocessor and 8 GB of RAM, ignoring network latency. The cryptographic key lengths of the Paillier and DGK cryptosystems are chosen according to NIST standards [33], which are valid until 2030. Table III.1.3 shows the parameters used for the implementation.

Table II.2.4: Parameters used in the implementation.

Parameter	Symbol	Value
Bit size of inputs	ℓ	2-30
Statistical security parameter	κ	112 bits
Paillier message space	n	2048 bits
DGK message space	u	31
DGK security parameter	t	224 bits
Error controller in NO07	φ	12

As [17] does not provide an analysis on φ, we implemented and analyzed their proposal. Figure II.2.2 shows various values of φ with their corresponding error rates. Furthermore, it presents run-time of the NO07 equality testing protocol with 25-bit inputs, and different φ values. As it is shown in Figure II.2.2, choosing $\varphi = 12$ makes the error probability negligible.

Figures II.2.3 and II.2.4 show the run-times of all the described secure equality testing protocols. Since VAT09, LT13, ST06, and NO07 are much more expensive regarding run-time than our protocols; we present their run-times separately in Figure II.2.4. As it is shown in Figure II.2.3, EQT-1 has the lowest run-time for inputs smaller than 20 bits. Figure II.2.4 shows the proposed protocols are computationally much more effi-

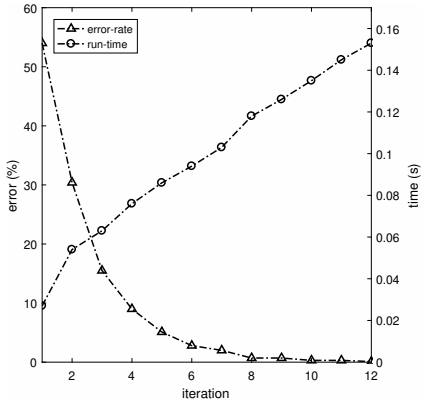

Figure II.2.2: The error-rates and run-times of NO07 for different values of φ.

cient than the state-of-the-art, as they outperform NO07, ST06, VAT09, and LT13 by 95%, 96%, 97%, and 99%, respectively for 25-bit inputs.

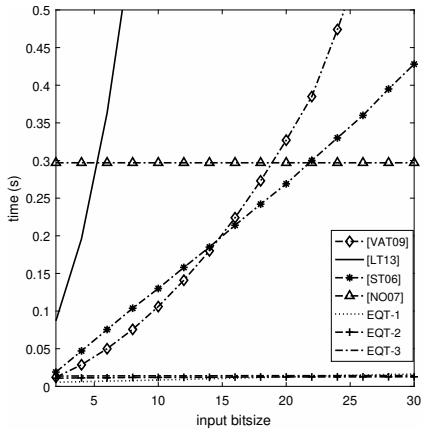

Figure II.2.3: Run-times of the equality testing protocols without data packing.

Notice that Figure II.2.3 shows the total run-time, which involves run-times of all operations including encryption and decryption. However, Table II.2.3 only presents the complexities of multiplication, exponentiation, and DGK zero-check, and it does not take into account the encryption and decryption costs.

2.5.3. APPLYING DATA PACKING

We observed that Paillier decryption dominates a significant portion of the total run-times of the protocols. For instance, it is shown in Table II.2.3 that EQT-3 is the most efficient protocol. However, in Figure II.2.3, EQT-3 does not demonstrate the same efficiency due to the cost of Paillier decryption. Data packing [34] can be used to mitigate such effect of Paillier decryption cost. Data packing reduces decryption cost since mul-

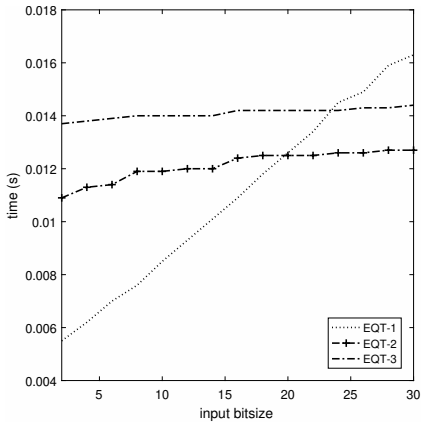

Figure II.2.4: Run-times of the equality testing protocols without data packing.

tiple messages packed in one ciphertext can be decrypted at once. However, data packing is applicable in the cases where there are multiple equality tests to be performed. This condition is realistic since in existing applications such as search algorithms many equality tests are needed. Since data packing uses the plaintext space of the encryption scheme, Paillier, efficiently, it also reduces the communication cost. Figures II.2.5 and II.2.6 show that the run-time and total data transmission of our equality testing protocols are reduced significantly after applying data packing. Notice that the results in Figure II.2.5 are matching our analysis in Table II.2.3, which does not take encryption and decryption into account. According to Figure II.2.5, EQT-3 after data packing outperforms VAT09, LT13, ST06, and NO07 by 99% for the inputs larger than 20 bits.

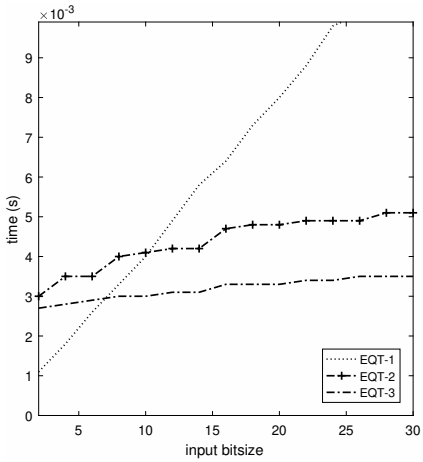

Figure II.2.5: Run-time of the equality testing protocols after data packing.

Table II.2.5 compares performance of the protocol based on run-time, communica-

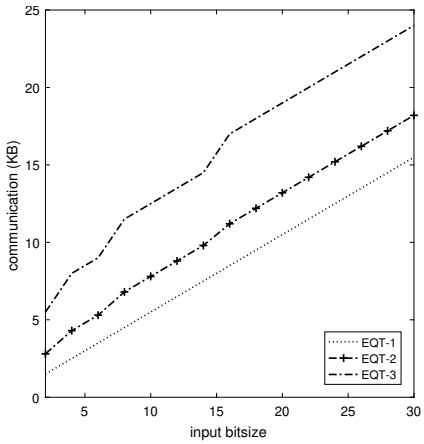

Figure II.2.6: Data transmission cost of the equality testing protocols after data packing.

tional round, and total data transmission for the inputs size of 20 bits. Table II.2.5 clearly shows the trade-off between communication and computation costs in all protocols.

Table II.2.5: Comparing EQT's performances ($\ell = 20, \varphi = 12, u = 31$).

Protocols	Run-time (sec)	Rounds	Data transmission (KB)
EQT-1	0.008	2	10
EQT-2	0.0048	3	13
EQT-3	0.0033	3	19

2.6. CONCLUSIONS

Testing equality of encrypted values is a building block in a number of cryptographic protocols such as searching in encrypted databases. In this work, we have investigated the state-of-the-art protocols and propose three new cryptographic protocols, which are significantly more efficient than the existing work regarding communication and computation. However, each protocol presented in this paper has its own advantages and disadvantages on run-time, bandwidth, and the number of rounds. Nevertheless, our analysis and experimental results support our claims in terms of efficiency compared to the state-of-the-art.

REFERENCES

[1] A. Sadeghi, T. Schneider, and I. Wehrenberg, *Efficient privacy-preserving face recognition*, in *Information, Security and Cryptology - ICISC 2009, 12th International Conference, Seoul, Korea, December 2-4, 2009, Revised Selected Papers* (2009) pp. 229–244.

[2] A. Jeckmans, A. Peter, and P. Hartel, *Efficient privacy-enhanced familiarity-based*

recommender system, in *Computer Security–ESORICS 2013* (Springer, 2013) pp. 400–417.

[3] Z. Yang, S. Zhong, and R. N. Wright, *Privacy-preserving classification of customer data without loss of accuracy,* in *Proceedings of the 2005 SIAM International Conference on Data Mining, SDM 2005, Newport Beach, CA, USA, April 21-23, 2005* (2005) pp. 92–102.

[4] C. Li, R. Lu, H. Li, L. Chen, and J. Chen, *Pda: a privacy-preserving dual-functional aggregation scheme for smart grid communications,* Security and Communication Networks **8**, 2494 (2015).

[5] 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.

[6] C. Hazay and T. Toft, *Computationally secure pattern matching in the presence of malicious adversaries,* J. Cryptology **27**, 358 (2014).

[7] R. Cramer, E. Kiltz, and C. Padró, *A note on secure computation of the moore-penrose pseudoinverse and its application to secure linear algebra,* in *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings* (2007) pp. 613–630.

[8] 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.

[9] G. Yang, C. H. Tan, Q. Huang, and D. S. Wong, *Probabilistic public key encryption with equality test,* in *Topics in Cryptology - CT-RSA 2010, The Cryptographers' Track at the RSA Conference 2010, San Francisco, CA, USA, March 1-5, 2010. Proceedings* (2010) pp. 119–131.

[10] Q. Tang, *Public key encryption supporting plaintext equality test and user-specified authorization,* Security and Communication Networks **5**, 1351 (2012).

[11] H. T. Lee, S. Ling, J. H. Seo, and H. Wang, *Semi-generic construction of public key encryption and identity-based encryption with equality test,* Inf. Sci. **373**, 419 (2016).

[12] K. Huang, R. Tso, and Y. Chen, *Somewhat semantic secure public key encryption with filtered-equality-test in the standard model and its extension to searchable encryption,* J. Comput. Syst. Sci. **89**, 400 (2017).

[13] L. Wu, Y. Zhang, K. R. Choo, and D. He, *Efficient and secure identity-based encryption scheme with equality test in cloud computing,* Future Generation Comp. Syst. **73**, 22 (2017).

[14] M. Ben-Or, S. Goldwasser, and A. Wigderson, *Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract)*, in *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA* (1988) pp. 1–10.

[15] R. Cramer, I. Damgård, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing* (Cambridge University Press, 2015).

[16] 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.

[17] T. Nishide and K. Ohta, *Multiparty computation for interval, equality, and comparison without bit-decomposition protocol*, in *Public Key Cryptography–PKC 2007* (Springer, 2007) pp. 343–360.

[18] B. Schoenmakers and P. Tuyls, *Efficient binary conversion for paillier encrypted values*, in *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings* (2006) pp. 522–537.

[19] I. Damgård and J. B. Nielsen, *Universally composable efficient multiparty computation from threshold homomorphic encryption*, in *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings* (2003) pp. 247–264.

[20] I. Damgård, M. Fitzi, E. Kiltz, J. B. Nielsen, and T. Toft, *Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation*, in *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings* (2006) pp. 285–304.

[21] J. Bar-Ilan and D. Beaver, *Non-cryptographic fault-tolerant computing in constant number of rounds of interaction*, in *Proceedings of the Eighth Annual ACM Symposium on Principles of Distributed Computing, Edmonton, Alberta, Canada, August 14-16, 1989* (1989) pp. 201–209.

[22] V. Kolesnikov and T. Schneider, *Improved garbled circuit: Free XOR gates and applications*, in *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations* (2008) pp. 486–498.

[23] V. Kolesnikov, A. Sadeghi, and T. Schneider, *Improved garbled circuit building blocks and applications to auctions and computing minima*, in *Cryptology and Network Security, 8th International Conference, CANS 2009, Kanazawa, Japan, December 12-14, 2009. Proceedings* (2009) pp. 1–20.

[24] 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.

[25] I. Damgård, M. Geisler, and M. Krøigaard, *A correction to 'efficient and secure comparison for on-line auctions',* IJACT **1**, 323 (2009).

[26] M. Nateghizad, Z. Erkin, and R. L. Lagendijk, *An efficient privacy-preserving comparison protocol in smart metering systems,* EURASIP Journal on Information Security **2016**, 1 (2016).

[27] O. Goldreich, *The Foundations of Cryptography - Volume 2, Basic Applications* (Cambridge University Press, 2004).

[28] 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.

[29] I. Damgård, M. Geisler, and M. Krøigaard, *A correction to 'efficient and secure comparison for on-line auctions',* IJACT **1**, 323 (2009).

[30] I. Damgård, M. Geisler, and M. Krøigaard, *Homomorphic encryption and secure comparison,* IJACT **1**, 22 (2008).

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

[32] T. Veugen, *Correction to "improving the DGK comparison protocol",* IACR Cryptology ePrint Archive **2018**, 1100 (2018).

[33] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, *Nist sp800-57: Recommendation for key management part 1: General (revised),* NIST, Tech. Rep (2007).

[34] J. R. Troncoso-Pastoriza, S. Katzenbeisser, M. Celik, and A. Lemma, *A secure multidimensional point inclusion protocol,* in *Proceedings of the 9th workshop on Multimedia & security* (ACM, 2007) pp. 109–120.

II.3 | A Communication-wise Efficient Equality Testing Protocol

Abstract

We propose a new secure equality testing (SET) protocol, namely SET-OT, for two-party setting by using a recently introduced Private Set Membership Protocol (PSM) based on Oblivious Transfer (OT) as a building block. We designed our equality test in such a way that the test result will not be revealed in clear text, which is desired in several cryptographic protocols. The advantage of using OT is that with the help of OT Extension (OTE) protocols, the cost of asymmetric operations per OT operations reduces when the number of OT executions increases. This makes our protocol competitive especially for the cases where the number of equality tests to be invoked is high. When the number of equality test increases, the time complexity of SET-OT converges to one asymmetric key decryption operation, this operation is the dominant part in terms of computational cost. SET-OT has a better performance in terms of the communication rounds and data transmission cost than state-of-the-art solutions: three communication rounds and 2.9 KB of data transmission are the communication costs of performing equality testing protocol for 20-bit string pairs. In addition to our complexity analysis, we also present test results to validate our claim on performance.

This chapter has been accepted as "SET-OT: A Secure Equality Testing Protocol Based on Oblivious Transfer", by F.Karakoç, M.Nateghizad and Z.Erkin to be published in the proceedings of 14th *International Conference on Availability, Reliability and Security*, 2019.

3.1. INTRODUCTION

Processing encrypted data has been a research topic in the field of privacy enhancing technologies and been deployed in several application domains [1–3]. Homomorphic encryption schemes are the tools that are widely used to enable performing linear operations directly over encrypted data. Other operations such as comparison and equality testing can be performed through secure multiparty computations (MPCs). However, using MPCs introduces significant amount of computation and communication overheads. Many improvements have been introduced to decrease the costs of the MPCs that are more frequently used in secure application.

In this paper, we introduce an efficient equality testing protocol. In our setting, a server holds encrypted pairs (x_1, x_2) and key manager has the keys. At the end of the protocol, the server obtains the equality relation between x_1 and x_2 in the encrypted form. Such a protocol has been used in the construction of several applications such as secure recommender systems [4, 5], where privacy-sensitive data are protected by means of encryption and the service provider performs operations on the encrypted data to provide its usual service. Our ambition is to improve the efficiency in terms of run-time, bandwidth and the number of rounds needed for such a protocol, particularly for the setting where the equality testing is needed in significantly large quantities.

Related Work. Yang et al. introduced a public key encryption scheme, namely PKwET, that supports to check the equality of two encrypted values [6]. Another protocol proposed by Tang allows only authorized parties to check the equality [7]. However, both solutions and the improved versions of PKwET [8–10] cannot be used for our scenario because the server that performs the equality test learns the result. Nishide and Ohta proposed an equality testing protocol in [11] that gives the correct result with 50% probability. Running the protocol many times to ensure the result is correct with a high probability increases the time and communication needs of the protocol. While the protocols introduced in [12, 13] are secure against active adversaries, they are inefficient in terms of the communication rounds and the amount of computation needed in a two-party setting. The protocols proposed in [14, 15] are based on Garbled Circuits, but their communication overhead is significantly high. There is a recent work that proposes using oblivious transfer [16]. To the best of our knowledge, that work is the only secure equality test protocol based on OT. However, the setting of this protocol differs from our setting. In the setting of [16], two parties have their private inputs, and at the end of the protocol, the parties only learn the shares of the result.

Our Contribution. We propose a new secure equality testing protocol, called SET-OT, that is efficient in terms of the number of communications rounds, computation time and the amount of data transferred between the two parties. To construct our protocol, we use the private set membership (PSM) protocol of Ciampi and Orlandi introduced in [17] as a building block where the PSM protocol is based on OT. The main property of this PSM protocol is that it does not reveal the membership relation in plaintext but allows the parties to learn the result in an encrypted format. Since our protocol uses Oblivious Transfer (OT) mainly, it gives better performance when the number of equality test operations increases by utilizing the Oblivious Extension Protocol (OTE) [18]. The number

of communication rounds of SET-OT is two if OT extension is not used, otherwise, the number of rounds becomes three. While $(1+2\lambda)$ asymmetric key decryption operations are needed to test two λ-bit strings, this cost reduces to $1+2\kappa/M$ per test if we test M pairs where κ is the security parameter. The amount of data transmitted between parties is approximately 2.9 KB to test 20-bit strings. The comparison of SET-OT's performance with the performances of the existing solutions are given in Table II.3.1.

Table II.3.1: Performance comparison of the protocols for the equality test of 20-bit strings. ([1]) Correctness parameter φ in [11] is taken as 12. ([2]) Performance result of SET-OT to test one pair. ([3]) Average performance result per pair when the protocol is executed to test 2800 pairs.

	# rounds	Time [ms]	Comm. [KB]
EQT-1 [19]	2	8	10
EQT-2 [19]	3	5	13
EQT-3 [19]	3	3	19
[12]	52	269	57.5
[1] [11]	36	300	14.5
[2] SET-OT	2	322	26.5
[3] SET-OT	3	3.8	2.9

The paper is organized as follows. In Section 3.2, we give the notation we use throughout the paper and provide the cryptographic primitives and protocols used as building blocks in our protocol. We introduce our protocol in Section 3.3. Then, we provide the security proof and the performance analysis of our protocol in Section 3.4 and 3.5, respectively. Finally, we conclude the paper in Section 1.6.

3.2. PRELIMINARIES

3.2.1. NOTATION

Throughout the paper, we use the notations that are listed in Table II.3.2.

3.2.2. SYSTEM SETTING

In this work, we assume that there are three parties: 1) clients, 2) server, and 3) key manager. Clients are the data owners. The server has computation resources to process a large volume of data. Key Manager (KM) generates a pair of public and private keys, then shares the public key with the other two parties. In this setting, the server and KM are both semi-trusted, which means they are honest to follow the protocols and do not collude, but they are curious to learn information that they are not entitled to. To protect privacy-sensitive data, clients encrypt their data with KM's public key before sending them to the server. To perform operations over encrypted data that demand decryption key, the server collaborates with KM such that KM cannot learn any private information after decrypting the encrypted data given from server. This setting helps to process the clients' encrypted data when they are off-line without violating the confidentiality of their data.

Table II.3.2: List of symbols

Symbol	Description
λ	Bit length of inputs
M	Number of pairs as inputs of SET
κ	Security parameter
η	Correctness parameter
s_r	Ciphertext bit length of RSA
s_p	Ciphertext bit length of Paillier
γ	SET result
$[x]$	Paillier encryption
$[x]_{asym}$	Asymmetric key encryption
$[\![x]\!]$,$[x]_{sym}$	Symmetric key encryption

3.2.3. PAILLIER ENCRYPTION SCHEME

Paillier Encryption Scheme is a public key cryptosystem introduced in [20]. The encryption operation of the scheme is defined as $\text{Enc}_{pk}(m,r) = g^m \times r^n \mod n^2$ for a given message m where $n = p \times q$, p and q are distinct prime numbers, r is a random number from $\mathbb{Z}_n^*$, g is a generator of $\mathbb{Z}_n^*$, (g,n) is the public key pk, and (p,q) is the private key sk. The encryption operation satisfies the additively homomorphic property that is the decryption of $Dec(\text{Enc}_{pk}(m_1,r_1) \times \text{Enc}_{pk}(m_2,r_2))$ equals to $(m_1+m_2) \mod n$. For a detailed description of the scheme the reader can refer to [20].

3.2.4. OBLIVIOUS TRANSFER

An Oblivious Transfer (OT) [21, 22] is a secure two-party protocol where party P_1^{OT} holds two strings and party P_2^{OT} wants to learn one of them according to her choice bit b, and at the end protocol P_1^{OT} learns nothing about the choice of P_2^{OT} and P_2^{OT} learns the string of her choice but nothing more. In the implementation of our protocol, we use the oblivious transfer whose steps are given below where the input of the P_2^{OT} (P_1^{OT}) is the choice bit b (two strings m_0 and m_1), the output of the P_2^{OT} (P_1^{OT}) is m_b ($\perp$) and $(d,(e,N))$ is the RSA key pair of P_1^{OT}:

1. P_2^{OT} generates three random strings x_0, x_1, and k, computes $v = (x_b + k^e) \mod N$ and sends x_0, x_1 and v to P_1^{OT}.
2. P_1^{OT} computes $k_0 = (v - x_0)^d \mod N$, $k_1 = (v - x_1)^d \mod N$, $m_0' = m_0 + k_0$, $m_1' = m_1 + k_1$ and sends m_0' and m_1' to P_2^{OT}.
3. P_2^{OT} computes $m_b = m_b' - k$ and outputs m_b .

In the rest of the paper, we use OT_ℓ^m notation to denote m execution of OT where the length of the strings is ℓ bits.

3.2.5. OBLIVIOUS TRANSFER EXTENSION

Ishai et al. introduced a protocol in [18] that extends OT_κ^κ to OT_ℓ^m. The steps of the OT extension protocol of Ishai et al. with the optimizations of [23, 24] as pictured in [25]

are as follows where the input of P_2^{OTE} (P_1^{OTE}) is the choice vector $b \in \{0,1\}^m$ (m pairs of ℓ-bit strings (m_0^i and m_1^i), $1 \le i \le m$), the output of the P_2^{OTE} (P_1^{OTE}) is m_b^i ($\perp$), κ is the security parameter, G is a PRG defined as $G : \{0,1\}^\kappa \longrightarrow \{0,1\}^m$ and H is a CRF defined as $H : \{0,1\}^\kappa \longrightarrow \{0,1\}^\ell$:

1. P_2^{OTE} generates κ random key pairs (k_0^i, k_1^i) where k_0^i and $k_1^i \in \{0,1\}^\kappa$ and $1 \le i \le \kappa$.
2. P_1^{OTE} generates a random vector $s \in \{0,1\}^\kappa$.
3. P_2^{OTE} and P_1^{OTE} run the OT Protocol κ times where in the i-th run P_2^{OTE} plays P_1^{OT} role in OT protocol with the input (k_0^i, k_1^i) and P_1^{OTE} plays P_2^{OT} role with the input i-th bit of s.
4. P_2^{OTE} constructs a $m \times \kappa$-bit random matrix $T = [t^1|...|t^\kappa]$ by computing $t^i = G(k_i^0)$ for $1 \le i \le \kappa$ where t^i is the i-th column of T and t_j denotes the j-th row of T and sends u^i to P_1^{OTE} where $u^i = t^i \oplus G(k_1^i) \oplus b$.
5. P_1^{OTE} constructs a $m \times \kappa$-bit matrix $Q = [q^1|...|q^\kappa]$ by computing $q^i = (s[i] \cdot u^i) \oplus G(k_i^{s[i]})$ for $1 \le i \le \kappa$ where q^i is the i-th column of Q and q_j denotes the j-th row of Q, computes $y_j^0 = m_j^0 \oplus H(q_j)$ and $y_j^1 = m_j^1 \oplus H(q_j \oplus s)$ for $1 \le j \le m$ and sends (y_j^0, y_j^1) to P_2^{OTE} for every $1 \le j \le m$. (Note that $q^i = (s[i] \cdot b) \oplus t^i$ and $q_j = (b[j] \cdot s) \oplus t_j$).
6. P_2^{OTE} computes $m_j^{b[j]} = y_j^{b[j]} \oplus H(t_j)$ for $1 \le j \le m$ and outputs $(m_{b[1]}^1, ..., m_{b[m]}^m)$.

3.2.6. CIAMPI AND ORLANDI'S PRIVATE SET MEMBERSHIP PROTOCOL

In the setting of Ciampi and Orlandi's Private Set Membership (PSM) Protocol [17], party P_2^{PSM} and party P_1^{PSM} hold a value x and a set of values Y, respectively. At the end of the protocol, P_2^{PSM} learns the membership only in an encrypted format, and P_1^{PSM} learns nothing about x. For our protocol, we use only the case that the number of elements in P_1^{PSM} is only one. We explain this protocol only for this case for simplicity. Let x_i and y_i denotes the i-th bit of x and y such that $x = x_\lambda x_{\lambda-1}...x_1$ and $y = y_\lambda y_{\lambda-1}...y_1$, respectively where λ is the bit length of the inputs x and y and y is the only element in Y. Assuming that input of the P_2^{PSM} (P_1^{PSM}) is a λ-bit string x (λ-bit string y), the output of P_2^{PSM} (P_1^{PSM}) is $F(\gamma^0)$ or $F(\gamma^1)$ depending on the equality relation between x and y ($\perp$), F can be any function, $E_k(.)$ is a symmetric encryption under the key k with the following property that the verification that a given ciphertext is in the range of $E_k(.)$ is efficient and the false positive probability is $2^{-\eta}$, the steps of the protocol are as follows:

1. P_1^{PSM} prepares pairs of values S_0^i and S_1^i for y_i where $\lambda \ge i \ge 1$ as follows:
   - chooses two symmetric keys k_{y_λ} and k_λ^* randomly and sets $S_{y_\lambda}^\lambda = k_{y_\lambda}$ and $S_{1-y_\lambda}^\lambda = k_\lambda^*$
   - chooses two symmetric keys $k_{y_\lambda y_{\lambda-1}...y_{i+1}y_i}$ and k_i^* randomly and sets

$$\begin{aligned} S_{y_i}^i &= \{E_{k_{y_\lambda y_{\lambda-1}...y_{i+1}}}(k_{y_\lambda y_{\lambda-1}...y_{i+1}y_i}), E_{k_{i+1}^*}(k_i^*)\} \\ S_{1-y_i}^i &= \{E_{k_{y_\lambda y_{\lambda-1}...y_{i+1}}}(k_i^*), E_{k_{i+1}^*}(k_i^*)\} \end{aligned}$$

for $i \in \{\lambda-1, \lambda-2, ..., 2, 1\}$.

- permutes the ciphertexts in $S^i_{y_i}$ and $S^i_{1-y_i}$ randomly.

2. $\mathrm{P}_1^{\mathrm{PSM}}$ sends $E_{k_{y_\lambda y_{\lambda-1}...y_1}}(F(\gamma^1))$ and $E_{k_1^*}(F(\gamma^0))$ to $\mathrm{P}_2^{\mathrm{PSM}}$ in random order.
3. $\mathrm{P}_2^{\mathrm{PSM}}$ learns corresponding $S^i_{x_i}$'s by running OT from $\mathrm{P}_1^{\mathrm{PSM}}$.
4. $\mathrm{P}_2^{\mathrm{PSM}}$ recovers only one of the keys $k_{y_\lambda y_{\lambda-1}...y_1}$ and k_1^* by decrypting the ciphertexts in the following way:
   - decrypts the ciphertexts in $S^{\lambda-1}_{x_{\lambda-1}}$ using $S^{\lambda}_{x_\lambda}$ as the key where the plaintext in the encryption domain is the key that will be used to decrypt the ciphertexts in $S^{\lambda-2}_{x_{\lambda-2}}$.
   - decrypts the ciphertexts in $S^i_{x_i}$ using the plaintext recovered from $S^{i+1}_{x_{i+1}}$ as the key to recover the key used in the next received message $S^{i-1}_{x_{i-1}}$.
5. $\mathrm{P}_2^{\mathrm{PSM}}$ decrypts the ciphertexts $E_{k_{y_\lambda y_{\lambda-1}...y1}}(F(\gamma^1))$ and $E_{k_1^*}F(\gamma^0))$ using the key recovered in Step 4 where only one of the plaintexts will be in the domain and this plantext will be equal to $F(\gamma^1)$ or $F(\gamma^0)$ according to the equality between x and y. $\mathrm{P}_2^{\mathrm{PSM}}$ outputs the result.

For a detailed definition of the protocol, the reader can refer to [17].

3.3. DEFINITION OF OUR PROTOCOL

The functionality of secure equality testing our protocol, SET-OT, computes is given in Definition 3.3.1.

Definition 3.3.1. The secure equality testing functionality $\mathscr{F}^{\mathscr{SET}} = (\mathscr{F}^{\mathscr{SET}}_{\mathrm{KM}}, \mathscr{F}^{\mathscr{SET}}_{\mathrm{S}})$ is

$$\begin{array}{rcl} \mathscr{F}^{\mathscr{SET}}_{\mathrm{S}} : (\{\{0,1\}^{s_p}\}^2, (\gamma^0, \gamma^1)) & \longrightarrow & \{0,1\}^{s_p} \\ (\mathsf{Enc}_{pk}(x_1), \mathsf{Enc}_{pk}(x_2), (\gamma^0, \gamma^1)) & \longmapsto & \begin{cases} \mathsf{Enc}_{pk}(\gamma^1) & \text{if } x_1 = x_2, \\ \mathsf{Enc}_{pk}(\gamma^0) & \text{if } x_1 \neq x_2. \end{cases} \\ & \text{and} & \\ \mathscr{F}^{\mathscr{SET}}_{\mathrm{KM}} : (\{\{0,1\}^{s_p}\}^2, (\gamma^0, \gamma^1)) & \longrightarrow & \perp \end{array}$$

where $\mathsf{Enc}_{pk}(x)$ is the encryption result of x under the public key pk whose corresponding private key is known by KM and S and KM are the server and the key manager, respectively.

We use the following primitives to construct our protocol:

- **The private set membership protocol in [17]**, $\Pi^{\in} = (\Pi^{\in}_{\mathrm{P}_1}, \Pi^{\in}_{\mathrm{P}_2})$, that realizes the functionality $\mathscr{F}^{\in} = (\mathscr{F}^{\in}_{\mathrm{P}_1}, \mathscr{F}^{\in}_{\mathrm{P}_2})$ securely in the semi-honest adversary model

$$\begin{array}{rcl} \mathscr{F}^{\in}_{\mathrm{P}_1} : (\{0,1\}^{\lambda} \times (\gamma^0, \gamma^1)) \times \{0,1\}^{\lambda} & \longrightarrow & \perp \\ & \text{and} & \\ \mathscr{F}^{\in}_{\mathrm{P}_2} : (\{0,1\}^{\lambda} \times (\gamma^0, \gamma^1)) \times \{0,1\}^{\lambda} & \longrightarrow & \{F(\gamma^0), F(\gamma^1)\} \\ (x_1, (\gamma^0, \gamma^1), x_2) & \longmapsto & \begin{cases} F(\gamma^1) & \text{if } x_1 = x_2, \\ F(\gamma^0) & \text{if } x_1 \neq x_2 \end{cases} \end{array}$$

where F is a function. The definition of the protocol is given in Section 3.2.6 for a special case we use in our protocol construction.

- **Paillier homomorphic encryption scheme [20]**
  $\mathsf{Asym} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ where Gen, Enc, and Dec are the key generation, encryption and decryption operations, respectively, $\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(m_1) \times \mathsf{Enc}_{pk}(m_2)) = (m_1 + m_2)$ and (pk, sk) is the public and secret key pair.

We give the steps of our protocol with Definition 3.3.2.

Definition 3.3.2. Our secure equality testing protocol,

$$\Pi^{\mathcal{SET}-\mathcal{OT}} = (\mathrm{S}^{\mathcal{SET}-\mathcal{OT}}, \mathrm{KM}^{\mathcal{SET}-\mathcal{OT}}),$$

proposed for the functionality in Definition 3.3.1 works as follows:
Common inputs: γ^0, γ^1, κ, η, λ
S's input: $\mathsf{Enc}_{pk}(x_1)$, $\mathsf{Enc}_{pk}(x_2)$, pk
KM's input: (pk, sk) (Paillier key pair)
S's output: $\mathsf{Enc}_{pk}(\gamma^1)$ or $\mathsf{Enc}_{pk}(\gamma^0)$ depending on the equality relation between x_1 and x_2
KM's output: $\perp$

1. S selects a random number r of length $\lambda + \kappa + 1$ and computes $\mathsf{Enc}_{pk}(x_1 - x_2 + r)$ using the additively homomorphic property of the encryption scheme Enc. Then sends the results to KM.

2. KM computes $(x_1 - x_2 + r)$ decrypting the ciphertext sent by S.

3. S and KM execute Π^{ϵ} where S runs $\Pi^{\epsilon}_{\mathrm{P}_2}$ taking the least significant λ bits of r as input, KM runs $\Pi^{\epsilon}_{\mathrm{P}_1}$ taking the least significant λ bits of $(x_1 - x_2 + r)$ as input and the common inputs are κ, η, $\mathsf{Enc}_{pk}(\gamma^0)$ and $\mathsf{Enc}_{pk}(\gamma^1)$. At the end of the execution, S obtains $\mathsf{Enc}_{pk}(\gamma^0)$ if x_1 is equal to x_2, $\mathsf{Enc}_{pk}(\gamma^1)$ otherwise.

Note that the private set membership of Ciampi et al. we use in Step 4 of our protocol can use OT or OT Extension protocol for oblivious transfer of S values so our protocol SET-OT can use either plain OT or the extension of OT. In the rest of the paper, we call our protocol as SET-OTE for the case of OT Extension is used.

3.4. SECURITY ANALYSIS

We analyze the protocol in the semi-honest security model [26]. In this security model, parties are assumed to follow the protocol without malicious activities. However, they are curious to learn more private data from other parties than they are entitled to. We use the simulation-based security proof paradigm given in [27] as a comparison of computation work-flow in the "real world" and the "ideal world".

In real world, each protocol consists of sub-protocols and multiple parties, where each party performs one or multiple sub-protocols. Let's denote π as our OT-based equality testing protocol. Since there are two parties: 1) key manager (KM) and 2) Server

(S); therefore, we can split π into two parts: π_{KM} and π_{S}. In our work, π_{S} takes two encrypted values $[x_1]$ and $[x_2]$ for equality test and outputs $[x_1 - x_2 + r]$ to π_{KM}. Then, π_{KM} decrypts the ciphertext given by π_{S}. Afterwards, π_{S} and π_{KM} jointly run Π^{ϵ}. In Π^{ϵ}, first, π_{KM} prepares S_0^i and S_1^i for each y_i where y equals to the least significant λ bits of r. Then, π_{S} obtains $S_{x_i}^i$, x is the least significant λ bits of $x_1 - x_2 + r$, and computes the encrypted result of the equality testing protocol. Assuming that an adversary $\mathscr{A}$ obtain the control of KM, then $\mathscr{A}$ has access to $x_1 - x_2 + r$, (S_0^i, S_1^i), and other values from OT protocol. And if S is corrupted by $\mathscr{A}$, he has access to r, $[x_1]$, $[x_2]$, $S_{x_i}^i$, encrypted result, and values from OT protocol.

In an ideal world, it is assumed that one of the corrupted parties uses a simulator to generate the outputs of the other parties. Such a setting is similar to the case, where the whole equality testing protocol is being performed with just one corrupted party. In the ideal world, an adversary $\acute{\mathscr{A}}$, who has control over KM, has access to S_0^i and S_1^i and garbage inputs from the simulator Sim_{S} instead of the correct result from S. Similarly, if $\acute{\mathscr{A}}$ has control over S, he has access to r, $[x_1]$, $[x_2]$ and garbage inputs from the simulator Sim_{KM}. The aim is to show that $\mathscr{A}$ can learn equal or negligibly more than $\acute{\mathscr{A}}$. This condition means operations in our equality testing protocol are computationally indistinguishable, and the protocol is simulation secure.

Definition 3.4.1. [28] It is said that the protocol Π securely computes the functionality $\mathscr{F} = (\mathscr{F}_1, \mathscr{F}_2)$ if there exist simulators Sim_1 and Sim_2 such that they are probabilistic polynomial-time algorithms,

$$\begin{aligned}&\{(Sim_1(1^\kappa, x, \mathscr{F}_1(x,y)), \mathscr{F}(x,y))\}_{\{x,y,\kappa\}} \overset{c}{\equiv}\\ &\{\mathrm{view}_{P_1}^{\Pi}(1^\kappa, x, y), \mathrm{output}^{\Pi}(1^\kappa, x, y)\}_{\{x,y,\kappa\}}\end{aligned}$$

and

$$\begin{aligned}&\{(Sim_2(1^\kappa, y, \mathscr{F}_2(x,y)), \mathscr{F}(x,y))\}_{\{x,y,\kappa\}} \overset{c}{\equiv}\\ &\{\mathrm{view}_{P_2}^{\Pi}(1^\kappa, x, y), \mathrm{output}^{\Pi}(1^\kappa, x, y)\}_{\{x,y,\kappa\}}\end{aligned}$$

where x and y arbitrary length bit strings with equal length and $\kappa \in \mathbb{N}$

Let denote the computation of $[x_1 - x_2 + r]$ as S_f and (S_0^i, S_1^i) as KM_f in our equality testing protocol. Let $\mathscr{F} = (\mathrm{S}_f, \mathrm{KM}_f)$, where $\mathscr{F}$ is the Probabilistic Polynomial-Time (PPT) functionality for our protocol, SET-OT. The View of the ith party, $i \in \{\mathrm{S}, \mathrm{KM}\}$ for the execution of SET-OT on $([a],[b];\phi)$ and security parameter n is denoted by $view_i^{SET-OT}([a],[b];\phi;n) = (w, r_i; m_1^i, \cdots, m_t^i)$, where $w \in \{([a],[b]),\phi\}$ depending on the value of i, r_i are the ith party internal random numbers, and m_j^i shows the jth message that is received by i^{th} party. Recall that KM does not have any initial input, thus its inputs is denoted as ϕ.

$output_i^{SET-OT}([a],[b];\phi;n)$ represents each party's output in SET-OT. We show the joint output of both parties as

$$\begin{aligned}output^{SET-OT} = (&output_1^{SET-OT}([a],[b];\phi;n),\\ &output_2^{SET-OT}([a],[b];\phi;n)).\end{aligned} \tag{3.1}$$

Definition 3.4.2. Assuming the semi-honest security setting, SET-OT computes $f = (\mathrm{S}_f, \mathrm{KM}_f)$ securely if there exits PPT algorithms Sim_{S} and Sim_{C} such that:

$$\{(Sim_{\mathrm{S}}(1^n, [a], [b], \mathrm{S}_f, f)\} \stackrel{c}{\equiv} \{(view_{\mathrm{S}}^f \\ ([a], [b]; \phi; n), output^f([a], [b]; \phi; n)\} \tag{3.2}$$

and

$$\{(Sim_{\mathrm{KM}}(1^n, \phi, \mathrm{KM}_f, f)\} \stackrel{c}{\equiv} \{(view_{\mathrm{KM}}^f \\ ([a], [b]; \phi; n), output^f([a], [b]; \phi; n)\} \tag{3.3}$$

The private set membership protocol is the main building block of SET-OT. This building block is proved to be simulation secure in [17]. Thus, in this section, we only analyze the security of the other parts of the SET-OT. As described in Section 3.3, SET-OT has three steps, and we only provide the security proof for the first two steps. In the first two steps, S sends an encrypted message to KM, but KM does not send any message to S in return. Therefore, proving the Equation 3.3 in SET-OT in unnecessary.

Theorem 7. *The protocol SET-OT is simulation secure and computes the functionality $\mathcal{F}^{\mathcal{SET}}$ in a privacy-preserving manner when* KM *is corrupted by adversary $\mathcal{A}$ in the semi-honest security setting.*

To achieve simulation security for SET-OT when KM is corrupted, we need to prove that $\mathcal{A}$ cannot distinguish between randomly generated messages from Sim_{S} and the correct versions from S. Recall that KM is given $[x_1 - x_2 + r]$ from S, which are under access of $\mathcal{A}$ who is holding the decryption key.

1. Sim_{S} generates and encrypts a random numbers $[R]$ as $[x_1 - x_2 + r]$, and then, sends it to KM.
2. $\mathcal{A}$ computes KM_f and sends the result to Sim_{S}.
3. Sim_{S} tosses random coins and randomly obtains S_x^i or S_y^i using OT protocol.

The outputs of the above simulation can be presented as $Sim_{\mathrm{S}}(1^n, \phi, S_f, f) = (\phi; [R], \acute{x}; ([\gamma]_{asym}, \phi))$, where the view is $view_{\mathrm{KM}}^f([x_1], [x_2]) = (\phi; [x_1 - x_2 + r]; ([\gamma]_{asym}, \phi))$. Moreover, $\mathcal{A}$ has access to the decryption key to open the given encrypted out of Sim_{S}. Therefore, we need to show that $\mathcal{A}$ cannot distinguish between (R) and $(x_1 - x_2 + r)$. The parameter, $x_1 - x_2$, is well masked with a $(\kappa + \lambda + 1)$-bit random number, r. This masking makes $\mathcal{A}$ unable to distinguish between (R) and $(x_1 - x_2 + r)$; therefore,

$$\{(Sim_{\mathrm{S}}(1^n, [a], [b], \mathrm{S}_f, f)\} \stackrel{c}{\equiv} \{(view_{\mathrm{S}}^f \\ ([a], [b]; \phi; n), output^f([a], [b]; \phi; n)\} \tag{3.4}$$

Based on the Theorem 9, we can claim that SET-OT protocol is simulation secure in the semi-honest security setting.

3.5. PERFORMANCE ANALYSIS

3.5.1. COMPLEXITY ANALYSIS

COMPLEXITY OF OBLIVIOUS TRANSFER.

P_1^{OT} and P_2^{OT} compute 2 RSA decryptions and 1 RSA encryption, respectively. P_1^{OT} sends $2s_r$-bit data to P_2^{OT} and receives $3s_r$-bit data from P_2^{OT}. In total, the operations performed by P_1^{OT} and P_2^{OT} and the total number of data transferred between the parties are 2 RSA decryptions and 1 RSA encryption and $5s_r$ bits, respectively. The number of rounds of OT protocol is 2.

COMPLEXITY OF OBLIVIOUS TRANSFER EXTENSION.

Let m be the number of OT operations P_2^{OTE} and P_1^{OTE} want to execute and ℓ be the bit length of the strings P_1^{OTE} will send obliviously. When we use OT Extension protocol instead of running m times OT, the computation and communication complexity becomes as follows. P_1^{OTE} and P_2^{OTE} execute κ times OT protocol playing P_2^{OT} and P_1^{OT} roles, respectively. P_1^{OTE} also computes $\kappa + 2m$ hash operations while P_2^{OTE} executes $2\kappa + m$ hashes. Thus, the computational complexity of the protocol is 2κ RSA decryptions, κ RSA encryptions and $3\kappa + 3m$ hash operations where P_1^{OTE} computes κ RSA encryption and $\kappa + 2m$ hashes and 2κ RSA decryptions and $2\kappa + m$ hashes are computed by P_2^{OTE}. In the protocol, P_1^{OTE} sends $2m\ell$-bit data to P_2^{OTE} and receives κm-bit data from P_2^{OTE} in addition to the data sent and received ($3\kappa s_r$ and $2\kappa s_r$) in κ OT executions. As a result total number of bits transferred in OTE protocol is $2m\ell + \kappa m + 5\kappa s_r$. The protocol can be executed in 3 rounds.

COMPLEXITY OF ONE RUN OF PSM PROTOCOL OF CIAMPI ET AL. WITH OT USAGE.

Let the length of the membership result be ℓ_r. P_1^{PSM} executes 3λ symmetric key encryptions to compute the S values in the protocol, 2 symmetric key encryptions to encrypt the possible membership results and runs λ times OT protocol as P_1^{OT}. Thus, total number of operations performed by P_1^{PSM} is 2λ RSA decryptions and $3\lambda + 2$ symmetric key encryptions. Similarly, P_2^{PSM} executes λ RSA encryptions and $2\lambda + 2$ symmetric key decryptions. The number of bits sent by P_1^{PSM} and P_2^{PSM} becomes $2\lambda s_r + 2\ell_r$ and $3\lambda s_r$, respectively. With the parallel execution of λ OT protocols, just two rounds are enough to execute the protocol.

COMPLEXITY OF MANY RUN OF PSM PROTOCOL OF CIAMPI ET AL. WITH OTE USAGE.

Let the number of protocol run be r, and the length of the membership result be ℓ_r. P_1^{PSM} executes $3r\lambda$ symmetric key encryptions to compute the S values in the protocol for r run, $2r$ symmetric key encryptions to encrypt the possible membership results, and runs OTE protocol for $r\lambda$ pairs of length $2(\kappa + \eta)$ as P_1^{OTE}. Thus, total number of operations performed by P_1^{PSM} is κ RSA encryptions, $\kappa + 2r\lambda$ hash computation and $3r\lambda + 2r$ symmetric key encryptions. Similarly, P_2^{PSM} executes 2κ RSA decryptions, $2\kappa + r\lambda$ hash computation and $2r\lambda + 2r$ symmetric key decryptions. The number of bits sent by P_1^{PSM} and P_2^{PSM} becomes $3\kappa s_r + 2r\lambda 2(\kappa + \eta) + 2r\ell_r$ and $2\kappa s_r + \kappa r\lambda$, respectively. Since P_1^{PSM} can send the encrypted result of the membership relation with the round in Step 5 of OTE protocol (Section 3.2.6), the number of rounds of this PSM protocol becomes three that is the number of rounds of OTE.

COMPLEXITY OF SET-OT TO TEST ONE PAIR.

Server (S) and key manager (KM) respectively perform one Paillier encryption and one Paillier decryption operations and run Ciampi et al. protocol as P_2^{PSM} and P_1^{PSM} for the input strings of length λ-bit and the membership result strings of length $s_p + \eta$. Thus, S performs $1 + \lambda$ asymmetric key encryptions and $2\lambda + 2$ symmetric key decryptions while KM executes $1 + 2\lambda$ asymmetric key decryptions and $3\lambda + 2$ symmetric key encryptions. S sends $s_p + 3\lambda s_r$-bit data that includes one Paillier encryption result and the data in the execution of Ciampi et al. protocol with the usage of OT. The number of bits sent by KM can be computed as $2\lambda s_r + 2(s_p + \eta)$ in a similar way. Thus the total amount data transmitted becomes $3s_p + 5\lambda s_r + 2\eta$ bits. S can send $[x_1 - x_2 + r_1]$ in Step 1 of OT protocol, which makes the number of rounds of the protocol two.

COMPLEXITY OF SET-OTE TO TEST MORE THAN κ/λ PAIRS.

When the number of OT execution for M pairs of length λ is bigger than the security parameter κ, it gives better performance result to use OT Extension protocol. In this case, S and KM respectively perform M Paillier encryption and M Paillier decryption operations and run Ciampi et al. protocol M times as P_2^{PSM} and P_1^{PSM} for the input strings of length λ-bit and the membership result strings of length $s_p + \eta$. Thus, S performs M asymmetric key encryptions, 2κ asymmetric decryptions, $2\kappa + M\lambda$ hash computation and $2M\lambda + 2M$ symmetric key decryptions while KM executes M asymmetric key decryptions, κ asymmetric encryptions, $\kappa + 2M\lambda$ hash computation and $3M\lambda + 2M$ symmetric key encryptions. S sends $Ms_p + 2\kappa s_r + \kappa M\lambda$-bit data that includes M Paillier encryption results and the data in the M execution of Ciampi et al. protocol. The number of bits sent by KM can be computed as $3\kappa s_r + 2M\lambda 2(\kappa + \eta) + 2M(s_p + \eta)$ in a similar way. The total amount of data transmitted in the execution of the protocol becomes $3s_p + 5\kappa s_r / M + 5\kappa\lambda + 4\lambda\eta + 2\eta$ per pair. The round complexity of SET-OT is equal to the round complexity of the underlying PSM protocol because the message transferred in Step 1 of our protocol can be sent in the same round of Step 2 in OT protocol that is executed in Step 3 of OT extension protocol.

COMPARISON OF OUR PROTOCOL WITH OTHER SOLUTIONS FROM LITERATURE.

Table II.3.3 and II.3.4 presents the comparison of our protocol with the existing solutions in terms of number of rounds and data amount transmitted in the execution of the protocol, respectively.

For the parameter values $\kappa = 112$, $s_p = 4096$, $s_r = 2048$, $M = 2800$, $\eta = 48$, and $\varphi = 12$, the communication complexity and the amount of data transmitted in the execution of the protocol for different λ values are shown in Table II.3.5. It can be obtained from the table that our protocol has the best performance in terms of the amount of data transmitted when many pairs are tested, and our protocol uses OT extension. In terms of round complexity, our solution is not the best one, but its round complexity can be considered at an acceptable level.

Table II.3.6 presents the performance comparison of our protocol with the existing solutions in terms of the number of cryptographic primitive calls. When we consider the dominant cryptographic primitive as the asymmetric key decryption in terms of computation time, our protocol execution time will converge to the execution time of one

Table II.3.3: Comparison of protocols in terms of number of rounds.

Protocol	# Rounds
[13]	2
[12]	$2\lambda + 2\lceil \log_2 \lambda \rceil + 2$
[11]	3φ
EQT 1 [19]	2
EQT 2 [19]	3
EQT 3 [19]	3
SET-OT	2
SET-OTE	3

Table II.3.4: Comparison of protocols in terms of data amount transmitted between parties.

Protocol	Transmitted data [KB]
[13]	$\lambda + 1$
[12]	$(6\lambda - 5)/2$
[11]	$(2\varphi + \lceil \log_2 \varphi \rceil + 1)/2$
EQT 1 [19]	$(\lambda + 2)/2$
EQT 2 [19]	$(\lambda + \lceil \log_2 \lambda \rceil)/2 + 2$
EQT 3 [19]	$(\lambda + 3\lceil \log_2 \lambda \rceil + 6)/2$
SET-OT	$(3s_p + 5\lambda s_r + 2\eta)/8192$
SET-OTE	$(3s_p + 5\kappa s_r/M + 5\kappa\lambda + 4\lambda\eta + 2\eta)/8192$

Table II.3.5: Communication complexities and amount of data in KB transmitted in the protocol executions for different λ values.

Protocol	Comm. comp.	$\lambda = 8$	$\lambda = 16$	$\lambda = 20$	$\lambda = 24$
[13]	$O(\lambda)$	9	17	21	25
[12]	$O(\lambda)$	21.5	45.5	57.5	69.5
[11]	$O(1)$	14.5	14.5	14.5	14.5
EQT 1 [19]	$O(\lambda)$	5	9	11	13
EQT 2 [19]	$O(\lambda)$	7.5	12	14.5	16.5
EQT 3 [19]	$O(\lambda)$	11.5	17	20.5	22.5
SET-OT	$O(\lambda)$	11.5	21.5	26.5	31.5
SET-OTE	$O(\lambda)$	2.3	3	3.4	3.8

asymmetric key decryption. In terms of asymmetric primitive call complexity, SET-OT with OTE usage is one of the best solutions as seen in the table.

Table II.3.6: Performance comparison in terms of cryptographic primitive calls.

Protocol	# of Operations	Complexity
[13]	$2\lambda+1$ asym. enc. 2 asym. dec.	$O(\lambda)$ asym.
[12]	5λ asym. enc. $2\lambda-1$ asym. dec.	$O(\lambda)$ asym.
[11]	$\lambda+2\varphi+1$ asym. enc. $2\varphi+1$ asym. dec.	$O(\lambda)$ asym.
EQT 1 [19]	$2\lambda+2$ asym. enc. 1 asym. dec.	$O(\lambda)$ asym.
EQT 2 [19]	$\lambda+2\log_2\lambda+5$ asym. enc. 2 asym. dec.	$O(\lambda)$ asym.
EQT 3 [19]	$\lambda+3\log_2\lambda+8$ asym. enc. 3 asym. dec.	$O(\lambda)$ asym.
SET-OT	$1+\lambda$ asym. key enc. $1+2\lambda$ asym. key dec. $2\lambda+2$ symm. dec. $3\lambda+2$ symm. enc.	$O(\lambda)$ asym. + $O(\lambda)$ symm.
SET-OTE	$1+\kappa/M$ asym. key enc. $1+2\kappa/M$ asym. key dec. $3\kappa/M+3\lambda$ hash comp. $3\lambda+2$ symm. enc. $2\lambda+2$ symm. dec.	$O(1)$ asym. + $O(\lambda)$ symm.

3.5.2. EXPERIMENTAL VERIFICATION

We implemented our protocol using C programming language and GMP library and run KM and S on the same machine where KM and S communicate in a TCP channel, for different item lengths and item pair counts on a standard PC with 2.26 GHz Intel Core i5 CPU and 6 GB RAM. In our implementation, we choose the parameters κ, η, s_p, s_r as 112-bit, 48-bit, 4096-bit and 2048-bit, respectively.

When we run the protocol for only one pair, we observed the performance result in Table II.3.7. According to the table, we see that the main computation cost is on the KM side because KM performs the asymmetric key decryption operations.

We also evaluated the performance of our protocol for the case that S holds more than one pair. This setting can be considered in real scenarios because most of the real world applications include equality test of many pairs. This case increases the number of OT operations in our protocol. To take advantage of OT Extension, we replaced the OT part in our implementation with OT Extension.

APPLICATION OF DATA PACKING

The dominant step in terms of computation complexity is the second step where one Paillier decryption is executed per item pair. Since the item lengths are relatively small

Table II.3.7: Performance results for one pair.

λ	# bytes transmitted	Comp time [ms] at KM	Comp time [ms] at S
8	11788	137.24	0.71
16	22028	260.97	1.28
20	27148	320.08	1.55
24	32268	386.19	2.18

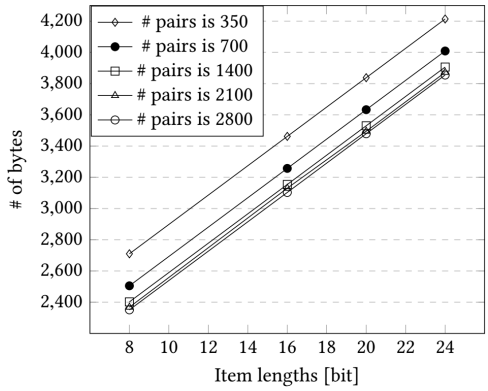

Figure II.3.1: Total # of bytes transmitted between KM and S for different # of pairs and item lengths when we use OT Extension and apply data packing.

than the Paillier modulus length, we can decrypt many ciphertexts by running only one Paillier decryption algorithm, with the help of the homomorphic property of the encryption scheme as follows: Let $[a]$ and $[b]$ be two ciphertexts, their lengths be ℓ bits and the modulus length of the encryption scheme be n where $n > 2\ell$. Then the decryption of $[a]^{2^\ell} \times [b]$ gives $2^\ell a + b$ where a and b can easily be recovered. In our implementation, since the length of the modulus is taken as 2048-bit and the plaintext length $(\kappa + \lambda + 1)$ is approximately 144-bit, we packed 14 ciphertexts into one ciphertext package that reduces the number of Paillier decryption operations by a factor of 14. The computation and communication complexities are given in Figure II.3.1 and II.3.2, respectively. As seen from the figure, utilizing data packing technique reduces the computation cost as expected.

The comparison of our protocol with the existing solutions in terms of communication and computation cost for $\lambda = 20$ is pictured in Figure II.3.3. It can be obtained from the figure that while our protocol is not the best solution to test only one pair, it has the best performance results if the number of pairs to be tested is high enough.

The main reason that makes our protocol one of the best solutions in literature is the usage of Oblivious Transfer Extension techniques. The extension techniques reduce the number of asymmetric key operations to the security parameter κ thus OT computation cost per OT operation is determined only by the number of symmetric key operations

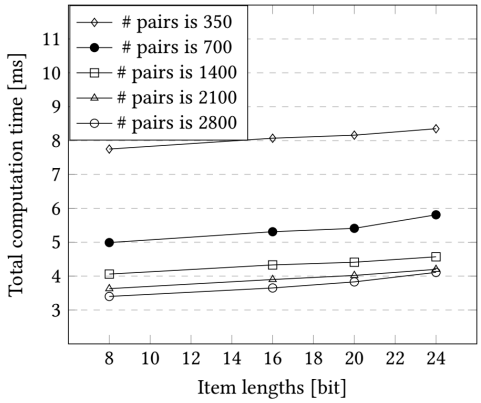

Figure II.3.2: Total computation time at KM and S for different # of pairs and item lengths when we use OT Extension and apply data packing.

when the number of OT operations is high enough.

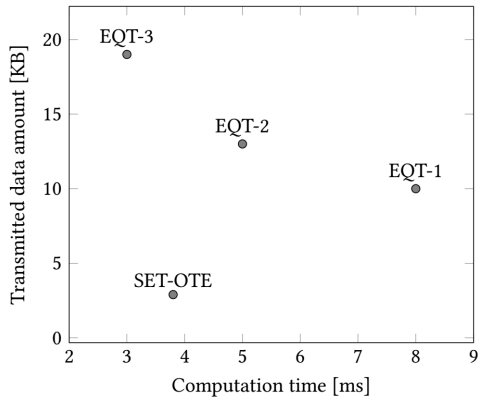

Figure II.3.3: Comparison of our protocol with protocols that have best performance results. Note that the performance of SET-OTE (SET-OT with OT Extension) was obtained running the protocol to test 1600 pairs and dividing the total amounts to the number of pairs.s

3.6. CONCLUSION

We proposed a secure equality testing protocol, SET-OT, by utilizing a recent Private Set Membership protocol that can be used as a building block for secure multi-party computation solutions. To the best of our knowledge, our protocol is the first protocol that converts a PSM protocol into a SET protocol. With the help of OT Extension technique, our protocol becomes one of the best protocols in terms of communication and computation complexities. Since the key manager performs most of the computation in our proposal, future work can be done to reduce the cost at the key manager side.

REFERENCES

[1] A. Sadeghi, T. Schneider, and I. Wehrenberg, *Efficient privacy-preserving face recognition,* in *Information, Security and Cryptology - ICISC 2009, 12th International Conference, Seoul, Korea, December 2-4, 2009, Revised Selected Papers* (2009) pp. 229–244.

[2] Z. Yang, S. Zhong, and R. N. Wright, *Privacy-preserving classification of customer data without loss of accuracy,* in *Proceedings of the 2005 SIAM International Conference on Data Mining, SDM 2005, Newport Beach, CA, USA, April 21-23, 2005* (2005) pp. 92–102.

[3] C. Li, R. Lu, H. Li, L. Chen, and J. Chen, *Pda: a privacy-preserving dual-functional aggregation scheme for smart grid communications,* Security and Communication Networks **8**, 2494 (2015).

[4] Z. Erkin, T. Veugen, and R. L. Lagendijk, *Privacy-preserving recommender systems in dynamic environments,* in *2013 IEEE International Workshop on Information Forensics and Security, WIFS 2013, Guangzhou, China, November 18-21, 2013* (2013) pp. 61–66.

[5] T. R. Hoens, M. Blanton, A. Steele, and N. V. Chawla, *Reliable medical recommendation systems with patient privacy,* ACM TIST **4**, 67:1 (2013).

[6] G. Yang, C. H. Tan, Q. Huang, and D. S. Wong, *Probabilistic public key encryption with equality test,* in *Topics in Cryptology - CT-RSA 2010, The Cryptographers' Track at the RSA Conference 2010, San Francisco, CA, USA, March 1-5, 2010. Proceedings*, Lecture Notes in Computer Science, Vol. 5985, edited by J. Pieprzyk (Springer, 2010) pp. 119–131.

[7] Q. Tang, *Public key encryption supporting plaintext equality test and user-specified authorization,* Security and Communication Networks **5**, 1351 (2012).

[8] K. Huang, R. Tso, and Y. Chen, *Somewhat semantic secure public key encryption with filtered-equality-test in the standard model and its extension to searchable encryption,* J. Comput. Syst. Sci. **89**, 400 (2017).

[9] H. T. Lee, S. Ling, J. H. Seo, and H. Wang, *Semi-generic construction of public key encryption and identity-based encryption with equality test,* Inf. Sci. **373**, 419 (2016).

[10] L. Wu, Y. Zhang, K. R. Choo, and D. He, *Efficient and secure identity-based encryption scheme with equality test in cloud computing,* Future Generation Comp. Syst. **73**, 22 (2017).

[11] T. Nishide and K. Ohta, *Multiparty computation for interval, equality, and comparison without bit-decomposition protocol,* in *Public Key Cryptography - PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography, Beijing, China, April 16-20, 2007, Proceedings*, Lecture Notes in Computer Science, Vol. 4450, edited by T. Okamoto and X. Wang (Springer, 2007) pp. 343–360.

[12] B. Schoenmakers and P. Tuyls, *Efficient binary conversion for paillier encrypted values,* in *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, Lecture Notes in Computer Science, Vol. 4004, edited by S. Vaudenay (Springer, 2006) pp. 522–537.

[13] 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*, Lecture Notes in Computer Science, Vol. 7966, edited by F. V. Fomin, R. Freivalds, M. Z. Kwiatkowska, and D. Peleg (Springer, 2013) pp. 645–656.

[14] V. Kolesnikov, A. Sadeghi, and T. Schneider, *Improved garbled circuit building blocks and applications to auctions and computing minima,* in *Cryptology and Network Security, 8th International Conference, CANS 2009, Kanazawa, Japan, December 12-14, 2009. Proceedings*, Lecture Notes in Computer Science, Vol. 5888, edited by J. A. Garay, A. Miyaji, and A. Otsuka (Springer, 2009) pp. 1–20.

[15] V. Kolesnikov and T. Schneider, *Improved garbled circuit: Free XOR gates and applications,* in *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, Lecture Notes in Computer Science, Vol. 5126, edited by L. Aceto, I. Damgård, L. A. Goldberg, M. M. Halldórsson, A. Ingólfsdóttir, and I. Walukiewicz (Springer, 2008) pp. 486–498.

[16] G. Couteau, *New protocols for secure equality test and comparison,* in *Applied Cryptography and Network Security - 16th International Conference, ACNS 2018, Leuven, Belgium, July 2-4, 2018, Proceedings*, Lecture Notes in Computer Science, Vol. 10892, edited by B. Preneel and F. Vercauteren (Springer, 2018) pp. 303–320.

[17] M. Ciampi and C. Orlandi, *Combining private set-intersection with secure two-party computation,* in *Security and Cryptography for Networks - 11th International Conference, SCN 2018, Amalfi, Italy, September 5-7, 2018, Proceedings*, Lecture Notes in Computer Science, Vol. 11035, edited by D. Catalano and R. D. Prisco (Springer, 2018) pp. 464–482.

[18] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank, *Extending oblivious transfers efficiently,* in *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, Lecture Notes in Computer Science, Vol. 2729, edited by D. Boneh (Springer, 2003) pp. 145–161.

[19] M. Nateghizad, T. Veugen, Z. Erkin, and R. L. Lagendijk, *Secure equality testing protocols in the two-party setting,* in *Proceedings of the 13th International Conference on Availability, Reliability and Security, ARES 2018, Hamburg, Germany, August 27-30, 2018*, edited by S. Doerr, M. Fischer, S. Schrittwieser, and D. Herrmann (ACM, 2018) pp. 3:1–3:10.

[20] 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*, Lecture Notes in Computer Science, Vol. 1592, edited by J. Stern (Springer, 1999) pp. 223–238.

[21] M. O. Rabin, *How to exchange secrets with oblivious transfer,* Cryptology ePrint Archive, Report 2005/187 (2005), `https://eprint.iacr.org/2005/187`.

[22] S. Even, O. Goldreich, and A. Lempel, *A randomized protocol for signing contracts,* Commun. ACM **28**, 637 (1985).

[23] V. Kolesnikov and R. Kumaresan, *Improved OT extension for transferring short secrets,* in *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, Lecture Notes in Computer Science, Vol. 8043, edited by R. Canetti and J. A. Garay (Springer, 2013) pp. 54–70.

[24] 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*, edited by A. Sadeghi, V. D. Gligor, and M. Yung (ACM, 2013) pp. 535–548.

[25] B. Pinkas, T. Schneider, and M. Zohner, *Scalable private set intersection based on OT extension,* ACM Trans. Priv. Secur. **21**, 7:1 (2018).

[26] O. Goldreich, *The Foundations of Cryptography - Volume 2, Basic Applications* (Cambridge University Press, 2004).

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

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

III

Comparison Protocol

III.1 | An Efficient Comparison Protocol

Abstract

In smart grids, providing power consumption statistics to the customers and generating recommendations for managing electrical devices are considered to be effective methods that can help to reduce energy consumption. Unfortunately, providing power consumption statistics and generating recommendations rely on highly privacy-sensitive smart meter consumption data. From past experience, we see that it is essential to find scientific solutions that enable the utility providers to provide such services for their customers without damaging customers' privacy. One effective approach relies on cryptography, where sensitive data is only given in the encrypted form to the utility provider and is processed under encryption without leaking content. The proposed solutions using this approach are very effective for privacy protection but very expensive in terms of computation and communication. In this paper, we focus on an essential operation for designing a privacy-preserving recommender system for smart grids, namely comparison, that takes two encrypted values and outputs which one is greater than the other one. We improve the state-of-the-art comparison protocol based on Homomorphic Encryption in terms of computation and communication by 56% and 25%, respectively by introducing algorithmic changes and data packing. As the smart meters are very limited devices, the overall improvement achieved is promising for the future deployment of such cryptographic protocols for enabling privacy enhanced services in smart grids.

This chapter has been published as "An efficient privacy-preserving comparison protocol in smart metering systems", by M.Nateghizad, Z.Erkin and R.L.Lagendijk in the *EURASIP Journal of Information Security*, 2016.

1.1. INTRODUCTION

Smart grids, as the next generation of power grid, are utilizing both communication technologies and information processing to monitor and manage power grids to enhance reliability, efficiency, and sustainability of power generation. One of the advantages of smart grids compared to traditional power grids is the ability to observe the power consumption of households in very short time intervals in the order of seconds to minutes. As a result of the fine-coarse data reporting, it is possible to provide power consumption statistics to the consumers, which might help to reduce the overall consumption by changing customer behavior, as pointed out in several works [1–5]. For example, Honebein *et al.* [6] defined people as the only true smart part of a smart grid; therefore monitoring, understanding, and promoting the end-users' roles from passive to active is considered as a fundamental action in smart grids. To this end, there are already several utility companies providing their customers devices and smart phone applications to monitor their real time consumption. Furthermore, one of the goals of the utility providers, balancing the supply and the demand, also known as demand response (DR), can be achieved more effectively if the utility provider can also provide statistics about the power usage in the surrounding area and generate personalized recommendations, for example to manage electrical devices like electric cars, heating systems, and ovens in the household [7].

Providing statistics on power consumption and generating personalized recommendations to inform customers are heavily dependent on the smart meter consumption readings. Unfortunately, these readings are highly privacy-sensitive [8–10]. The utility provider can use the readings from the smart meters for other purposes, misuse them or even transfer them to other entities without the consent of the customers. As seen in many cases, privacy is considered to be a big challenge for using smart meters to the fullest extent, e.g. enabling personalized services such as generating recommendations.

In this paper, we assume that the utility provider generates statistics and recommendations for the customers so that the customers can adjust the electrical devices for the most cost-effective and environmentally friendly manner. To achieve this, we rely on cryptography, which provides us tools to create Privacy by Design algorithms. For instance, there are already a number of studies for computing bills and aggregating data [11–14]. The main idea in this research line is to provide only the encrypted power consumption to the utility provider and enable processing the encrypted data without decrypting any sensitive information. This way, the utility provider cannot access to the content but at the same time can perform the algorithms required for the service. Unfortunately, the cryptographic algorithms for this purpose are expensive in terms of computation and communication, which mostly require smart meters to be involved in the computation [15–18]. Since the smart meters are very limited devices, improving the efficiency of the cryptographic algorithms is a challenge.

We address the efficiency problem of a fundamental operation, namely comparison, which is required to design any recommender system. In our setting, the encrypted consumption readings are collected from the customers by an aggregator and the utility provider has the decryption key. For privacy reasons, the aggregator cannot transfer the data directly to the utility provider but can co-operate with the utility provider to generate recommendations. One important step in the system is to compare values, which are

only available in the encrypted form. More precisely, the aggregator has two encrypted values, and it needs to know which one is greater than the other one without revealing their contents to anyone including itself.

There are numerous comparison protocols designed for comparing encrypted values [15, 16, 18]. In this paper, we improve the state-of-the-art comparison protocol that relies on homomorphic encryption in terms of run-time by 56% by introducing algorithmic changes. Furthermore, we also reduce communication cost of the protocol by 25% by deploying data packing [19, 20]. Together, these improvements increase the overall efficiency of the comparison protocol with encrypted inputs, bringing smart meters one step closer to run privacy-preserving cryptographic protocols based on homomorphic encryption.

Note that a secure comparison protocol with encrypted values is needed in many applications, not only for generating recommendations, like face recognition [17], fingercode authentication [21], and K-means clustering [22]. Therefore, the protocol we improved in this paper provide a significant performance improvement for other applications as well.

1.2. PRELIMINARIES

In this section, we describe the application setting, the security assumptions, and the cryptographic tools used in this work. We also present the symbols and their descriptions in Table III.1.1.

Table III.1.1: Symbols and their descriptions.

Symbol	Description	Symbol	Description
a, b	Secret inputs	h	Uniformly random number
s_k	Secret key	z	The integer $2^\ell + a - b$
p_k	Public key	z_ℓ	The most significant bit of z
$\mathscr{E}$	Encryption function	d	Masked version of z, $d = z + r$
$\mathscr{D}$	Decryption function	$\hat{r}$	The integer $r \bmod 2^\ell$
m	Plaintext	$\acute{d}$	The integer $d \bmod 2^\ell$
$\mathbb{Z}_\eta$	Paillier message space	δ	Uniformly random bit
$\mathbb{Z}u$	DGK message space	$[.]$	Paillier encryption
$\in_R X$	A random number in X	s	The integer $1 - 2 \cdot \delta$
ℓ	Bit length of secret inputs	$[\![.]\!]$	DGK encryption
κ	Security parameter	ρ	Number of ciphertext that
λ_η	Carmichael function		can be packed into one
r	$\kappa + \ell$-bit random number		Paillier ciphertext
$\mathrm{ord}(\alpha)$	The smallest positive integer	$\hat{d}$	packed Paillier ciphertexts
	x such that $\alpha^x = 1 \bmod n$	λ	Comparison output
$/$	Integer division	$\Psi(x)$	$\lfloor x/2^\ell \rfloor$

1.2.1. APPLICATION SETTING

In our application setting, we define three roles: 1) smart meters installed at the households, 2) a data aggregator, and 3) a utility provider. Smart meters measure, encrypt, and send consumers' power consumption to the data aggregator, which collects and analyzes encrypted power consumption. Then, the utility provider generates recommendations for its customers by running a cryptographic protocol with the data aggregator. The output of the cryptographic protocol, which depends on the purpose of the recommender system, is in the encrypted form, thus it is not available neither to the data aggregator nor to the utility provider. The output is then revealed to the customer by using another protocol, secure decryption, which is explained in [23].

1.2.2. SECURITY MODEL

The proposed protocol in this work is built on the semi-honest adversarial model, where the data aggregator and the utility provider are honest in the sense that they faithfully follow the designed protocol but will try to infer information from the protocol execution transcript. This assumption is realistic since companies are expected to properly perform required services mentioned in the service level agreement, when engaging in a collaboration. We assume that the utility provider is the only party holding the private keys, while the smart meters and the data aggregator have the public keys for the encryption schemes. We assume that neither party colludes.

1.2.3. HOMOMORPHIC ENCRYPTION

In this work, we rely on two additively homomorphic cryptosystems, Paillier [24] and DGK (Damgård, Geislet and Krøigaard) [15]. An additively homomorphic encryption scheme preserves certain structure that can be exploited to process ciphertexts without decryption. Given $\mathscr{E}_{pk}(m_1)$ and $\mathscr{E}_{pk}(m_2)$, a new ciphertext whose decryption yields the sum of the plaintext messages m_1 and m_2 can be obtained by performing a certain operation over the ciphertexts: $\mathscr{D}_{sk}(\mathscr{E}_{pk}(m_1)) \otimes (\mathscr{E}_{pk}(m_2)) = m_1 + m_2$.

Consequently, exponentiation of any ciphertext with a public value yields the encrypted product of the original plaintext and the exponent: $\mathscr{D}_{sk}(\mathscr{E}_{pk}(m)^e) = e \cdot m$.

1.2.4. PAILLIER CRYPTOSYSTEM

The Paillier encryption function for a given message $m \in \mathbb{Z}_\eta$ is defined as follows:

$$c = \mathscr{E}_{pk}(m, \tau) = g^m \cdot \tau^\eta \mod \eta^2, \tag{1.1}$$

where η is the product of two distinct large prime numbers p and q, ciphertext $c \in \mathbb{Z}^*_{\eta^2}$, $\tau \in_R \mathbb{Z}^*_\eta$ and g is a generator of order η. The decryption function is,

$$\frac{L_\eta(c^{\lambda_\eta} \mod \eta^2)}{L_\eta(g^{\lambda_\eta} \mod \eta^2)} \mod \eta = m, \tag{1.2}$$

where λ_η is the Carmichael value that is the smallest positive integer such that $\{\forall a \in \mathbb{Z}^*_\eta : a^{\lambda_\eta} \equiv 1 \,(\mathrm{mod}\ \eta)\}$ and $L_\eta(x) = \frac{x-1}{\eta}$. The public key is (g, η) and the private key is λ_η.

The homomorphic property can be shown as below:

$$\begin{aligned}&\mathscr{D}_{sk}((\mathscr{E}_{pk}(m_1)) \times (\mathscr{E}_{pk}(m_2)) = \mathscr{D}_{sk}(g^{m_1} \cdot \tau_1^{\eta} \times g^{m_2} \cdot \tau_2^{\eta})\\ &= \mathscr{D}_{sk}(g^{m1+m2} \cdot (\tau_1 \cdot \tau_2)^{\eta}) = \mathscr{D}_{sk}(\mathscr{E}_{pk}(m_1 + m_2)) = m_1 + m_2 \,. \end{aligned} \tag{1.3}$$

1.2.5. DGK CRYPTOSYSTEM

We also use the DGK cryptosystem [15, 25], which is used in constructing cryptographic protocols [17, 23] for its efficiency due to its small message size. For generating the public and the private keys, there are three parameters: k, t, and ℓ, where $\ell < t < k$. The process of key generation is as follows:

1. Choose two distinct t-bit prime numbers v_p, v_q.
2. Construct two distinct prime numbers p and q, where $v_p|(p-1)$ and $v_q|(q-1)$ such that $n = pq$ is a k-bit RSA modulus.
3. Choose u, the smallest possible prime number but greater than $\ell + 2$.
4. Choose a random r that is a $2.5t$-bit integer [15].
5. Choose g and h such that $\text{ord}(g) = uv_pv_q$ and $\text{ord}(h) = v_pv_q$.

The public and the private keys are $pk = (n, g, h, u)$ and $sk = (p, q, v_p, v_q)$, respectively. The encryption of a plaintext $m \in \mathbb{Z}_u$ is given as follows:

$$c = \mathscr{E}_{pk}(m, r) = g^m \cdot h^r \mod n \,. \tag{1.4}$$

To decrypt the ciphertext one can build a look-up table for all $m \in \mathbb{Z}_u$ values and obtain m from $c^{v_p} \mod p = (g^{v_p})^m \mod p$. However, DGK scheme can efficiently check whether a ciphertext is an encryption of zero or not. To achieve this, we check whether $c^{v_pv_q} \mod n = 1$ or more efficiently we only need to prove that $c^{v_pv_q} \mod p = 1$ or $c^{v_pv_q} \mod q = 1$, since $u < p$.

In the rest of the paper, we denote the ciphertext of a message m by $[m]$ for the Paillier cryptosystem and $[\![m]\!]$ for the DGK.

1.3. SECURE COMPARISON PROTOCOL WITH SECRET INPUTS

In this section, we describe the state-of-the-art secure comparison protocol (SCP), which takes two encrypted inputs and outputs the greater one in the encrypted form. SCP based on the DGK construction introduced in [25] is one of the widely-used comparison protocols due to its efficiency. The DGK comparison protocol is a sub-protocol in the SCP, where each party possesses a secret but plaintext value. The sub-protocol also uses the DGK cryptosystem for efficiency reasons.

The comparison protocol in [25] is modified and used by Erkin *et al.* in [17], and Veugen proposed an improved DGK comparison protocol (IDCP) in [18]. In the following, we describe the SCP construction.

For the sake of simplicity, we use the names Alice and Bob as the data aggregator and the utility provider, respectively. We assume that Bob has the secret key sk and Alice has access to two encrypted values, $[a]$ and $[b]$, and wants to know if $a < b$.

Initially, Alice computes $[z] = [2^\ell + a - b] = [2^\ell] \cdot [a] \cdot [b]^{-1}$, and then obtains the result of comparison as follows:

$$[z_\ell] = [2^{-\ell} \cdot (z - (z \bmod 2^\ell))] = ([z] \cdot [z \bmod 2^\ell]^{-1})^{2^{-\ell}}, \tag{1.5}$$

where $[z_\ell]$ is the most significant bit of $[z]$ and the result of comparison. If $z_\ell = 1$ then we have $a > b$, and otherwise $a < b$. A more efficient method of computing $[z_\ell]$ is based on the IDCP, where we can compute $z_\ell = \lfloor z/2^\ell \rfloor$ and $[a < b] = [1 - z_\ell] = [1] \cdot [z_\ell]^{-1}$, but we still need to compute $[z \bmod 2^\ell]$. A more detailed explanation regarding computation of $[z_\ell]$ is provided in the following sections.

1.3.1. COMPUTING $[z \bmod 2^\ell]$

Notice that Alice has access only to $[z]$, and interaction with Bob, who has the private key, is needed to compute modulo reduction, $[z \bmod 2^\ell]$. However, Alice cannot give $[z]$ directly to Bob since this value reveals information on the difference of a and b. Therefore, Alice masks $[z]$ using a random value as follows:

$$[d] = [z + r] = [z] \cdot [r], \tag{1.6}$$

where r is a $(\kappa + \ell)$-bit uniformly random number and κ is a security parameter. After masking, Alice sends $[d]$ to Bob to perform modulo reduction, where Bob first decrypts $[d]$, then computes $\acute{d} = d \bmod 2^\ell$ and sends $[\acute{d}]$ and $[d/2^\ell]$ back to Alice. Subsequently, to obtain $[z \bmod 2^\ell]$, Alice computes $[\tilde{z} \bmod 2^\ell] = [\acute{d} - r \bmod 2^\ell] = [\acute{d}] \cdot [r \bmod 2^\ell]^{-1}$.

Note that $z \bmod 2^\ell = \tilde{z} \bmod 2^\ell$ if $\acute{d} > r \bmod 2^\ell$. When $\acute{d} < r \bmod 2^\ell$, an underflow occurs, and Alice has to add 2^ℓ to $[\tilde{z}]$ to make the value positive again. Therefore, Alice needs to determine whether $\acute{d} > r \bmod 2^\ell$ or not. This is achieved by computing an encrypted value, $[\lambda]$, which shows the relation between $\acute{d}$ and $r \bmod 2^\ell$. Then, Alice can perform following computation to obtain $[z \bmod 2^\ell]$:

$$[z \bmod 2^\ell] = [\tilde{z} + \lambda 2^\ell] = [\tilde{z}] \cdot [\lambda]^{2^\ell}. \tag{1.7}$$

Alice can obtain $[z_\ell]$ by using Equation 1.5. $[z_\ell]$ can be computed more efficiently as follow:

$$[z_\ell] = [\Psi(z)] = [\Psi(d)] \cdot [\Psi(r)]^{-1} \cdot [\lambda]^{-1} \tag{1.8}$$

where $\Psi(x) = \lfloor x/2^\ell \rfloor$. For computing $[\lambda]$, we run a secure comparison protocol with private inputs as described in the following section.

1.3.2. COMPUTING $[\lambda]$

This protocol outputs an encrypted bit, which shows whether $\acute{d} > \hat{r} = r \bmod 2^\ell$ or not [17]. However, different than the original problem of comparing encrypted a and b, in this protocol Alice and Bob possess $\hat{r}$ and $\acute{d}$ in plaintext, respectively. Based on this setting, the IDCP for computing $[\lambda]$ securely works as follows:

1. Bob sends a bitwise encryption of his input, $[\![\acute{d}_0]\!], \ldots, [\![\acute{d}_{\ell-1}]\!]$, to Alice.

2. Alice chooses uniformly random bit δ, where $\delta \in \{0,1\}$. Then she computes $s = 1-2\cdot\delta$ and $[\![c_i]\!]$ as follows,

$$[\![c_i]\!] = [\![\acute{d}_i - \hat{r}_i + s + 3\sum_{j=i+1}^{\ell-1} \acute{d}_j \oplus \hat{r}_j]\!]$$
$$= [\![\acute{d}_i]\!]\cdot[\![\hat{r}_i]\!]^{-1}\cdot[\![s]\!]\cdot\Big(\prod_{j=i+1}^{\ell-1}[\![\acute{d}_j \oplus \hat{r}_j]\!]\Big)^3, \tag{1.9}$$

where $[\![\acute{d}_j \oplus \hat{r}_j]\!] = [\![\acute{d}_j]\!]\cdot[\![\hat{r}_j]\!]\cdot[\![\acute{d}_j]\!]^{-2\cdot\hat{r}_j}$, and $i = 0,\ldots,\ell-1$.

3. Alice blinds each $[\![c_i]\!]$ with a uniformly random $h_i \in_R \mathbb{Z}_u^*$ such that

$$[\![e_i]\!] = [\![c_i \cdot h_i]\!] = [\![c_i]\!]^{h_i}, \tag{1.10}$$

then permutes $[\![e_i]\!]$ and sends them to Bob. Note that if $c_t = 0$, where $t \in \{0,\ldots,\ell-1\}$ then $e_t = 0$ as well.

4. Bob checks whether there is a zero among $[\![e_i]\!]$ values. If none of the $[\![e_i]\!]$ values are encrypted zero then he sets $\tilde{\lambda} = 0$, otherwise $\tilde{\lambda} = 1$. Then he encrypts $\tilde{\lambda}$ and sends $[\tilde{\lambda}]$ to Alice.

5. Alice corrects $[\tilde{\lambda}]$ to obtain $[\lambda]$ as follows:

$$[\lambda] = \begin{cases} [\tilde{\lambda}] & \text{if } s = 1 \\ [1]\cdot[\tilde{\lambda}]^{-1} & \text{if } s = -1 \end{cases}$$

After obtaining $[\lambda]$, Alice computes $[z \mod 2^\ell]$ and $[z_\ell]$ based on Equations 1.7 and 1.5 respectively.

EFFICIENT PRIVACY-PRESERVING COMPARISON PROTOCOL (EP-PCP)

In this section, we describe a new version of the original SCP based on the DGK construction, which is significantly more efficient in terms of run-time and communication cost.

1.3.3. PROPOSED COMPARISON PROTOCOL

Complexity analysis and experimental results reveal that the XOR operation in computing $[\![c_i]\!]$, in Equation 1.9, has a significant impact on the overall efficiency of the DGK comparison protocol for the following two reasons:

1. Computing XOR is computationally expensive, since $[\![\hat{r} \oplus \acute{d}]\!] = [\![\hat{r}]\!]\cdot[\![\acute{d}]\!]\cdot[\![\acute{d}]\!]^{-2\cdot\hat{r}}$. Veugen [18] proposed a more efficient technique of computing XOR, where $[\![\hat{r} \oplus \acute{d}]\!] = [\![\acute{d}]\!]$ when $\hat{r} = 0$; otherwise, $[\![\hat{r} \oplus \acute{d}]\!] = [\![1]\!]\cdot[\![\acute{d}]\!]^{-1}$ (Recall that Alice and Bob have access to values $\hat{r}$ and $\acute{d}$, respectively and Alice is computing XOR). Thus, if $\hat{r}$ equals to 1, one multiplication and one exponentiation with negative exponent should be computed over DGK ciphertexts, which affects the performance of DGK comparison protocol significantly.

2. Since the equation that involves XOR is computed during the protocol with encrypted inputs, it is not possible to introduce pre-computation for $[\![c_i]\!]$ to obtain a more efficient protocol.

Table III.1.5 shows that computing $[\![c_i]\!]$ constitutes 70% of the overall run-time of the IDCP for Alice.

Based on these two facts, we propose a more efficient way of computing $[\![c_i]\!]$, which does not rely on the original XOR computation. The value $[\![c_i]\!]$ can be re-written as follows:

$$[\![c_i]\!] = [\![\acute{d}_i - \hat{r}_i + s + \sum_{j=i+1}^{\ell-1} (\acute{d}_j \cdot 2^j \; - \; \hat{r}_j \cdot 2^j)]\!] \,. \tag{1.11}$$

Alice computes Equation 1.11 in three steps:

1. Bob computes $[\![t_i]\!] = [\![\acute{d}_i + \sum_{j=i+1}^{\ell-1} \acute{d}_j \cdot 2^j]\!]$, and sends $[\![t_i]\!]$ to Alice,

2. Alice computes $[\![v_i]\!] = [\![s - \hat{r}_i - \sum_{j=i+1}^{\ell-1} \hat{r}_j \cdot 2^j]\!]$, and

3. Alice computes $[\![c_i]\!]$ as follows,

$$[\![c_i]\!] = [\![t_i + v_i]\!] = [\![t_i]\!] \cdot [\![v_i]\!] \,. \tag{1.12}$$

Note that Alice can pre-compute $[\![v_i]\!]$ and factor '3' is not needed in the computation of $[\![c_i]\!]$. After computing all $[\![c_i]\!]$ values, Alice masks each $[\![c_i]\!]$ and sends masked values to Bob, where he checks if any of the given masked $[\![c_i]\!]$ is zero, then generates $[\tilde{\lambda}]$, and sends it to Alice. She corrects $[\tilde{\lambda}]$ based on value s to obtain $[\lambda]$, computes Equation 1.7, and 1.5 to obtain $[z_\ell]$ as in the original protocol. Note that we compare $2\acute{d}$ and $2\hat{r}$ instead of $\acute{d}$ and $\hat{r}$ respectively for technical reasons explained in the following section.

CORRECTNESS PROOF OF COMPUTING $[\![c_i]\!]$

In this section we prove the correctness of generating $[\![c_i]\!]$ by Equation 1.12. In order to do that, we check if Equation 1.12 generates encrypted zero in the same conditions as the Equation 1.9. Table III.1.2 shows the values of c_i computed based on the EPPCP and the IDCP, which are denoted as c_i^E and c_i^I respectively. Table III.1.2 analyzes the existence of zero in c_i^E generated based on the s, $\acute{d}$, $\hat{r}$, and $S_{i+1} = \sum_{j=i+1}^{\ell-1} (\acute{d}_j \cdot 2^j \; - \; \hat{r}_j \cdot 2^j)$ values. Based on this table, the value of c_i^{IDCP} can be zero in two conditions, where $\{\acute{d} < \hat{r}$, $s = 1$, $\acute{d}_i = 0$ and $\hat{r}_i = 1\}$ and $\{\acute{d} > \hat{r}$, $s = -1$, $\acute{d}_i = 1$ and $\hat{r}_i = 0\}$. However, c_i^E generates zero in more conditions than c_i^I does. For instance, if $\{S_{i+1} = 2, s = -1, \acute{d} > \hat{r}$ and $\acute{d}_i = \hat{r}_i = 1\}$, then $c_i^E = 0$. Table III.1.2 shows that values of c_i^I can be zero in the conditions 4 and 5; however, c_i^E values are zero in the conditions 3, 4, 5, and 6 based on the assumed values of S_{i+1} for each condition. We note that if $\acute{d}_1\acute{d}_0 * \hat{r}_1\hat{r}_0 = 2$ and $\acute{d}_i = \hat{r}_i$ for $2 \leqslant i \leqslant \ell - 1$, then the value of c_0^E becomes zero. To fix this problem, we compare $2\acute{d}$ and $2\hat{r}$ instead of $\acute{d}$ and $\hat{r}$. Therefore, Equation 1.12 does not generate zero in the conditions 3 and 6. Furthermore, for the comparison protocol to work when $\acute{d} = \hat{r}$, we compare $3\acute{d}$ and $3\hat{r}+1$ instead of $\acute{d}$ and $\hat{r}$ respectively, as suggested similarly in [17].

Table III.1.2: Different conditions based on $s, \acute{d}$ and $\hat{r}$.

Condition	$\acute{d} > \hat{r}$	s	$\acute{d}_i$	$\hat{r}_i$	S_{i+1}	c_i^E	c_i^I
1	True	1	0	1	0	nonzero	nonzero
2	True	1	1	0	−2	nonzero	nonzero
3	True	−1	0	1	2	zero	nonzero
4	True	−1	1	0	0	zero	zero
5	False	1	0	1	0	zero	zero
6	False	1	1	0	−2	zero	nonzero
7	False	−1	0	1	2	nonzero	nonzero
8	False	−1	1	0	0	nonzero	nonzero

1.3.4. DATA PACKING

According to Table III.1.5, Paillier decryption of $[d]$ (Equation 1.6) dominates more than 62% of the comparison protocol execution time at Bob side. We decrease the run-time of Paillier decryption by employing data packing similar to [19, 20]. The main idea behind data packing is to efficiently use the message space of the Paillier cryptosystem that is much larger than the values to be compared.

Assume that z and r are ℓ and $\ell+\kappa$-bit integers, respectively. Then, $[d] = [z+r]$ is a $(\ell+\kappa+1)$-bit integer. Let the message space of the Paillier cryptosystem be $\eta = pq$, then Alice packs $\rho = \eta / \lfloor(\ell+\kappa+1)\rfloor$ into one Paillier message as follows:

$$[\hat{d}] = \sum_{j=0}^{\rho-1} [d]_j^{(2^{\ell+\kappa+1})^j}, \tag{1.13}$$

and sends $[\hat{d}]$ to Bob. Then, Bob computes $\mathscr{D}_{sk}([\hat{d}])$, unpacks ρ different values and performs modulo reduction on each unpacked value.

Employing the data packing technique not only reduces the number of very expensive Paillier decryption to be performed, but also decreases the number of encrypted messages to be transmitted.

1.4. PERFORMANCE ANALYSIS

In this section, we analyze the number of operations over ciphertexts, since they are computationally expensive compared to operations on the plaintext and dominate the protocol execution run-time, and provide experimental results for run-time performance. For this purpose, we implemented the EPPCP using C++ and SeComLib [26] library, on a Linux machine running Ubuntu 14.04 LTS, with 64-bit microprocessor and 8 GB of RAM. The experiments are repeated for 10,000 comparisons. Table III.1.3 provides more information about parameters and their corresponding values in our implementation.

Table III.1.6 shows the computational complexity of the original DGK comparison protocol, the IDCP, and the EPPCP. Note that the number of multiplications and exponentiations are regarding the computation of $[\![c_i]\!]$. According to the Table III.1.6, the original DGK comparison protocol suffers from its high computational complexity regarding the number of multiplications and exponentiations over ciphertexts. Veugen

Table III.1.3: Parameters and their values used in the implementation.

Parameter	Symbol	Value
Bit size of inputs	ℓ	25 bits
Security parameter	κ	40 bits
Paillier message space	η	2048 bits
DGK message space	n	2048 bits
Number of $[d]$ packed into one Paillier ciphertext	ρ	31

[18] presented two improvements to decrease the computational cost of the DGK comparison protocol, namely an efficient method to compute XOR and an algorithm to mask less $[\![c_i]\!]$, which results in a lower number of exponentiations with positive exponent. However, according to Table III.1.6, the new technique of computing XOR have a slight impact on the overall number of multiplications and exponentiations. Moreover, Table III.1.5 shows that computation of $[\![e_i]\!]$ takes 15% of the protocol run-time in Alice (the improvement for computing $[\![e_i]\!]$ [18] is not applied in the implementation); therefore, even a significant improvement over computation of $[\![e_i]\!]$ does not provide a significant influence on the overall run-time.

Table III.1.4: Computational complexity of original DGK [17, 25], the IDCP and the EPPCP.

Function	Original DGK	IDCP	EPPCP
Encryption	$1_{Paillier} + \ell_{DGK}$	$1_{Paillier} + \ell_{DGK}$	$1_{Paillier} + \ell_{DGK}$
Decryption	$1_{Paillier}$	$1_{Paillier}$	$(\frac{1}{\rho})_{Paillier}$
DGK zero-check	ℓ	ℓ	ℓ
Multiplication	$\ell(\ell+2)$	$\sim \frac{\ell(\ell+11)}{4}$	ℓ
Exponentiation(+)	ℓ	ℓ	0
Exponentiation(−)	$\frac{\ell(\ell+1)}{2}$	$\sim \frac{\ell(\ell+3)}{4}$	0

Table III.1.6 shows that the computational complexity of computing $[\![c_i]\!]$ in the EPPCP is decreased to ℓ multiplications over ciphertexts, and there is no exponentiation with positive or negative exponent. According to Table III.1.7, this low computational complexity results in 91% decrease in computation of $[\![c_i]\!]$ compared to the IDCP. This improvement also reduces the run-time of all computations performed by Alice by 64%.

Table III.1.5: Run-time performance for several steps of the IDCP.

Function	Time (second)	Overal computation (%)
	Alice	
Computing $[\![c_i]\!]$	15	70
$[\![e_i]\!] \leftarrow$ Masking $[\![c_i]\!]$	3.15	15
Other	3.15	15
	Bob	
DGK zero-check	27.3	38
Paillier decryption	44.4	62
Total	93	

Table III.1.6: Computational complexity of original DGK [17, 25], the IDCP and the EPPCP.

Function	Original DGK	IDCP	EPPCP
Encryption	$1_{Paillier} + \ell_{DGK}$	$1_{Paillier} + \ell_{DGK}$	$1_{Paillier} + \ell_{DGK}$
Decryption	$1_{Paillier}$	$1_{Paillier}$	$(\frac{1}{\rho})_{Paillier}$
DGK zero-check	ℓ	ℓ	ℓ
Multiplication	$\ell(\ell+2)$	$\sim \frac{\ell(\ell+11)}{4}$	ℓ
Exponentiation(+)	ℓ	ℓ	0
Exponentiation(−)	$\frac{\ell(\ell+1)}{2}$	$\sim \frac{\ell(\ell+3)}{4}$	0

Table III.1.5 also shows that Paillier decryption dominates 62% of the IDCP run-time by Bob. According to the Table III.1.7, by deploying data packing the run-time of the Paillier decryption and all Bob's computations are decreased by 85% and 53%, respectively.

Table III.1.8 shows the running times of the Paillier decryption (PD), computation of c_i, and the total run-time (online phase) of both the EPPCP and the IDCP for different key sizes. It shows that the EPPCP achieves better efficiency compared to the IDCP for the large key sizes.

According to Table III.1.9, running EPPCP 10,000 times takes 41 seconds, where it takes 93 seconds for the IDCP. Table III.1.9 also shows that pre-computation phase takes more time in EPPCP as a result of the new method of computing $[\![c_i]\!]$, which allows performing more initial computations before run-time. The communication cost between Alice and Bob is decreased by 25% in EPPCP because of using data packing technique.

Table III.1.7: Run-time performance of the several steps of the EPPCP and the improvements compared to the IDCP.

Function	Time (second)	Improvement (%)
	Alice	
Computing $[\![c_i]\!]$	1.40	91
$[\![e_i]\!] \leftarrow$ Masking $[\![c_i]\!]$	3.15	0
Other	3.15	0
	Bob	
DGK zero-check	27.3	0
Paillier decryption	6.40	85
Total	41.4	

Table III.1.8: performance of the IDCP and the EPPCP for different key lengths. In this table, protocol is denoted as Prot., Pillier decryption as PD, run-time as RT.

	512 bits			1024 bits			2048 bits		
Prot.	c_i	PD	RT	c_i	PD	RT	c_i	PD	RT
IDCP	2.7	0.8	6.9	5.9	6.1	21.5	15.0	44.4	93.0
EPPCP	0.1	0.5	4.2	0.3	2.0	13.2	1.4	6.4	41.4
%	96%	37%	39%	95%	67%	39%	91%	85%	55%

SECURITY AND PRIVACY OF COMPARISON PROTOCOL

In this section, we provide a security sketch of the proposed privacy-preserving comparison protocol in the semi-honest model. For a more elaborate security proof, we refer readers to [25].

Table III.1.9: Overall performance of the IDCP and the EPPCP.

Protocol	Run-time(sec)	Pre-computation(sec)	communications
IDCP	93	7.4	40k
EPPCP	41.4	13.8	30k
Improvement	+56%	−87%	+25%

As mentioned before, smart meters encrypt the power consumption using the Paillier cryptosystem, which is semantically secure under the decisional composite residuosity assumption (for more information about the security of Paillier cryptosystem we refer reader to [24]), thus Alice (data aggregator) has only encrypted values. Here, we show that not only does Bob (utility provider) not learn anything about the given encrypted

values but also Alice does not learn any information about encrypted output of the algorithm at the end of the proposed comparison protocol.

Alice computes together with Bob $[z \bmod 2^\ell]$ without revealing any information about $[z]$ to him. Since this value reveals information on the distance between a and b; therefore, Alice masks $[z]$ by adding a random value, $[d] = [z+r]$, and sends $[d]$ to Bob instead of $[z]$. Since r is a uniformly random $(k+\ell)$-bit value, $[z]$ is statistically indistinguishable from $[d]$ to Bob.

Bob sends back $[d \bmod 2^\ell]$ to Alice in the encrypted form, which means she cannot learn any information about the content of $[z]$, but only $[z \bmod 2^\ell]$. Then Alice sends $[\![e_i]\!]$ values, which are the masked and the permuted $[\![c_i]\!]$ values, to Bob who checks the existence of an encrypted zero among given $[\![e_i]\!]$. Therefore, Bob only receives a list of uniformly random values. Moreover, using a binary random value s through computation of $[\![c_i]\!]$ prevents Bob from drawing any conclusions about the result of the comparison by checking $[\![e_i]\!]$. Since Alice is not authorized to know the result of the comparison, Bob only sends the encrypted value of $\tilde{\lambda}$, $[\tilde{\lambda}]$, to Alice. Then she can only correct the $[\tilde{\lambda}]$ based on s to obtain $[\lambda]$ and compute $[z_\ell]$.

1.5. CONCLUSION

Comparing consumers' power consumption profiles is a necessary part of smart grids for a number of services including generating personalized recommendations. Since personal profiles contain private information about consumers' power consumption, privacy-preserving approaches should be considered. One of the most effective approaches is based on using cryptographic tools that enable processing encrypted data. Unfortunately, secure and privacy-sensitive versions of such services are computationally expensive, which hinders the deployment of such protocols in practice. In this paper, we investigated a vital operation that is invoked numerous times during many algorithms, namely comparison, and improve its performance significantly by introducing algorithmic changes and deploying data packing. By doing so, we improve the efficiency of the state-of-the-art secure comparison protocol based on homomorphic encryption. More precisely, we reduce the run-time of computations by the data aggregator and the utility provider by 64% and 52%, respectively. In terms of overall performance, the proposed comparison protocol is faster than state-of-the-art by 56% and the communication cost is reduced by 25%. This improvement in performance leads to a more practical comparison protocol that can be used for designing privacy-preserving protocols.

REFERENCES

[1] G. P. Verbong, S. Beemsterboer, and F. Sengers, *Smart grids or smart users? involving users in developing a low carbon electricity economy,* Energy Policy **52**, 117 (2013).

[2] I. Lampropoulos, G. Vanalme, and W. L. Kling, *A methodology for modeling the behavior of electricity prosumers within the smart grid,* in *Innovative Smart Grid Technologies Conference Europe (ISGT Europe), 2010 IEEE PES* (IEEE, 2010) pp. 1–8.

[3] T. Jackson, *Motivating sustainable consumption: a review of evidence on consumer*

behaviour and behavioural change: a report to the Sustainable Development Research Network (Centre for Environmental Strategy, University of Surrey, 2005).

[4] S. Van Dam, *Smart energy management for households* (TU Delft, Delft University of Technology, 2013).

[5] A. Spagnolli, N. Corradi, L. Gamberini, E. Hoggan, G. Jacucci, C. Katzeff, L. Broms, and L. Jönsson, *Eco-feedback on the go: Motivating energy awareness,* Computer **44**, 38 (2011).

[6] P. C. Honebein, R. F. Cammarano, and C. Boice, *Building a social roadmap for the smart grid,* The Electricity Journal **24**, 78 (2011).

[7] V. Giordano, F. Gangale, G. Fulli, M. S. Jiménez, I. Onyeji, A. Colta, I. Papaioannou, A. Mengolini, C. Alecu, T. Ojala, *et al.*, *Smart Grid projects in Europe: lessons learned and current developments* (Citeseer, 2011).

[8] J. Liu, Y. Xiao, S. Li, W. Liang, and C. Chen, *Cyber security and privacy issues in smart grids,* Communications Surveys & Tutorials, IEEE **14**, 981 (2012).

[9] P. McDaniel and S. McLaughlin, *Security and privacy challenges in the smart grid,* IEEE Security & Privacy , 75 (2009).

[10] F. G. Marmol, C. Sorge, O. Ugus, and G. M. Pérez, *Do not snoop my habits: preserving privacy in the smart grid,* Communications Magazine, IEEE **50**, 166 (2012).

[11] K. Birman, M. Jelasity, R. Kleinberg, and E. Tremel, *Building a secure and privacy-preserving smart grid,* ACM SIGOPS Operating Systems Review **49**, 131 (2015).

[12] Z. Erkin and G. Tsudik, *Private computation of spatial and temporal power consumption with smart meters,* in *Applied Cryptography and Network Security* (Springer, 2012) pp. 561–577.

[13] K. Kursawe, G. Danezis, and M. Kohlweiss, *Privacy-friendly aggregation for the smart-grid,* in *Privacy Enhancing Technologies* (Springer, 2011) pp. 175–191.

[14] C. Efthymiou and G. Kalogridis, *Smart grid privacy via anonymization of smart metering data,* in *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on* (IEEE, 2010) pp. 238–243.

[15] I. Damgard, M. Geisler, and M. Kroigard, *A correction to'efficient and secure comparison for on-line auctions',* International Journal of Applied Cryptography **1**, 323 (2009).

[16] T. Veugen, F. Blom, S. de Hoogh, and Z. Erkin, *Secure comparison protocols in the semi-honest model,* (2014).

[17] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, *Privacy-preserving face recognition,* in *Privacy Enhancing Technologies* (Springer, 2009) pp. 235–253.

[18] T. Veugen, *Improving the dgk comparison protocol,* in *Information Forensics and Security (WIFS), 2012 IEEE International Workshop on* (IEEE, 2012) pp. 49–54.

[19] J. R. Troncoso-Pastoriza, S. Katzenbeisser, M. Celik, and A. Lemma, *A secure multidimensional point inclusion protocol,* in *Proceedings of the 9th workshop on Multimedia & security* (ACM, 2007) pp. 109–120.

[20] T. Bianchi, A. Piva, and M. Barni, *Composite signal representation for fast and storage-efficient processing of encrypted signals,* Information Forensics and Security, IEEE Transactions on **5**, 180 (2010).

[21] M. Barni, T. Bianchi, D. Catalano, M. D. Raimondo, R. D. Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, F. Scotti, and A. Piva, *Privacy-preserving fingercode authentication,* in *Multimedia and Security Workshop, MM&Sec 2010, Roma, Italy, September 9-10, 2010* (2010) pp. 231–240.

[22] M. Beye, Z. Erkin, and R. L. Lagendijk, *Efficient privacy preserving k-means clustering in a three-party setting,* in *2011 IEEE International Workshop on Information Forensics and Security, WIFS 2011, Iguacu Falls, Brazil, November 29 - December 2, 2011* (2011) pp. 1–6.

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

[24] P. Paillier, *Public-key cryptosystems based on composite degree residuosity classes,* in *Advances in cryptology EUROCRYPT 99* (Springer, 1999) pp. 223–238.

[25] I. Damgård, M. Geisler, and M. Krøigaard, *Efficient and secure comparison for online auctions,* in *Information security and privacy* (Springer, 2007) pp. 416–430.

[26] C. S. Group, *SeComLib secure computation library,* (2013).

IV

Secure Searching and Retrieval

IV.1 | Secure Index-Based Search Protocols

Abstract

It is astonishing to see more and more services built on user-oriented data, providing numerous tools to improve one's daily life. Nowadays, data collected from numerous sources is being used to monitor daily activities, i.e., monitoring patients. These innovations allow for more cost-efficient and scalable solutions. Nevertheless, these types of services can pose a threat to the privacy of individuals due to the possibility of leaking highly privacy-sensitive data. Therefore, it is essential to design such systems in a privacy-preserving manner. Inspired by a real-life project in the health-care domain, we propose to secure the data using encryption, while enabling the involved parties to run queries directly on this encrypted data. A vital component of such a system is searching for specific data entries within a large dataset. In this work, we present two cryptographic protocols that complete such a query by creating an encrypted vector in a simulation secure way. These vectors consist of a 1 for intended database entry, whereas other items would be represented as a 0. By creating index tables before the execution of the queries, it has become possible to execute a search query with high performance. As we show in our analyses, it takes less than one second to find the matching encrypted data-entry within a database with 100K records. Our proposal is generic, can be applied to several application domains, and practically compared to similar works.

This chapter has been published as "Efficient Index-based Search Protocols for Encrypted Databases", by M.Nateghizad, Z.Erkin, and R.L.Lagendijk in the proceedings of 15th *International Joint Conference on e-Business and Telecommunications*, pages 436-447, 2018.

1.1. INTRODUCTION

A real-life problem motivates the work presented in this paper: A hospital wants to monitor its patients using off-the-shelf smart devices [1] that measure, among other things, a patient's weight, ECG, blood pressure, and blood sugar level. These devices connect to the smartphone of a patient, who needs to be monitored on a daily basis. A mobile application then sends the measurements to the central server of the vendor. Afterward, the hospital can use a web-based application to check the measurements for any particular patient. The primary reasons to use such a system are straight-forward: scalability and cost reduction [2].

Unfortunately, the whole system relies on the assumption that the vendor is trustworthy and it has a secure method to protect against both internal and external attackers. All recent incidents show that this is not yet the case [3]. Currently, it is possible for a hacker to break into the data servers and steal privacy-sensitive medical data, such an attacker can either be a malicious employee or someone who makes a genuine mistake [4]. It is essential to propose a system where sensitive data are protected while enabling medical institutions to monitor their patients remotely.

In order to create a secure system with this layout, we propose to encrypt measurements directly on the smartphone of the user before sending them to the vendor. We aim to encrypt the data using a homomorphic encryption scheme which enables data processing while encrypted, without revealing the content of data to the vendor or any third party. More precisely, what is needed for the above system is to identify a patient, or a group of patients, with specific conditions, e.g., people with high blood pressure within a particular time period. Provided that we require semantic security, it is challenging to find all the data for given conditions, since it requires searching through the encrypted database.

Searching in encrypted databases has been a challenge for researchers for many decades. Proposed solutions vary in the cryptographic tools used for encrypting data. Examples include: schemes built on attribute-based encryption [5], homomorphic encryption [6] and special constructions such as Oblivious RAMS [7, 8]. The focus has been on improving efficiency; the current state-of-the-art is not as practical as searching within a plaintext database where different techniques can be used to speed up the search function (e.g., creating hash tables). Therefore, there is a need for further research to achieve higher efficiency.

Inspired by the medical application, in this work, we assume that patients have a smartphone, which can collect measurements from one or more smart devices. The data is then sent to the vendor's storage unit, which can either be a local or cloud-based database. More interestingly, a patient might utilize different smart devices from different vendors. The hospital that wants to monitor a specific patient, or a group of patients, should be able to retrieve the related data without leaking information to any of the vendors.

This application setting is challenging for three reasons: 1) we want to enable hospitals to retrieve data on sophisticated queries, 2) we also assume multiple devices from different vendors and 3) the amount of data collected from the patients is significantly large. To achieve our goal, we need to identify the data entries, which are all encrypted, that match the query provided by the hospital. More precisely, we assume that there is a

global (virtual) database with encrypted entries from all devices. Given that database, we want to create an encrypted, binary vector such that a vector element is 1 for corresponding database entry and 0, for all other tuples. Given such an encrypted binary vector, it is possible to build numerous services such as i) generating statistics (i.e., counting, averaging), ii) data aggregation, and iii) private data retrieval.

To obtain such a vector, we present two cryptographic protocols for secure searching, IBSvI and IBSvII. In IBSvI, we propose a computation-wise efficient searching protocol. In IBSvII, the computation of generating the encrypted binary vector is performed in one party. However, IBSvII introduces more computational overhead. These protocols rely on creating index tables and updating them with each input received from a device. The index tables are then used later to execute queries and find specific database entries at significantly lower cost. Our proposal has several advantages over existing works: 1) our protocols are designed for numerical data, in contrast to current work that relies on exact match, 2) our proposal supports conjunction queries with “AND", 3) our proposal is simulation secure; it leaks no private information including search pattern and access pattern to the involved entities, and 4) our protocols enable generating statistics from encrypted data based on the given conditions.

1.2. RELATED WORK

Ostrovsky and Goldreich [7, 8] introduced ORAM where it is possible to evaluate any query, while the access pattern is kept hidden. ORAM lets users upload their private data to a remote storage in encrypted form, and still have random access to their data in a secure way. However, ORAM allows users to access only one entry at a time with a logarithmic number of communication rounds for each read. Moreover, in ORAM, users should know the location of the data that they are looking for in the database. Later works [9–11] proposed more efficient ways of searching by using weaker security models. Song et al. [9] introduced a private key based searching that is communication-wise more efficient than ORAM. The secure searching in [9] is based on generating and storing a two-layer ciphertext in the remote storage unit. Although the introduced encryption scheme by Song et al. is proven to be secure, the searching procedure reveals the access pattern. Similarly, Stefanov et al. [12] combined a secure search and ORAM, where the keywords are kept confidential, but the search protocol still reveals the access pattern to the remote storage. To improve the searching performance, as one of the limitations of Song et al. [9], Goh [10], and Chang and Mitzenmacher [11] proposed two new secure searching protocols with indexing. They constructed an index table alongside each set of data in the remote storage. The remote storage uses the index tables to find the matching data instead of checking every single encrypted data. Although [10, 11] are efficient, their proposals leak access pattern to the untrusted parties.

Curtmola et al. [13] also presented a semantically secure search by using asymmetric data encryption, which is capable of finding desired data in sub-linear search time. In [13], each user constructs an index, which includes every possible data that can appear in a query, then the index table is deterministically encrypted and outsourced to the remote storage. To perform a search, a user constructs a query that contains a token that is a deterministic function of the search data and sends the token to the remote storage. Then, the remote storage unit searches for the specific data in each set. Al-

though the proposed searching technique is fast, it has two limitations: 1) users cannot update the index table of their data unless they generate the index table again, and 2) the searching technique still reveals the access pattern. To overcome the challenge of updating index table in [13], Kamara et al. [14] introduced an improvement, which enables updating the index table. However, the problem of revealing the access pattern is not addressed in that work. [15] proposes an efficient and secure search that supports top-k similarity search over encrypted data by using a random traversal algorithm. However, in [15], users cannot evaluate their queries, but only the data owner. [16] presents an efficient search technique over encrypted data that uses Bloom filter as the indexing technique. Although using Bloom filter introduces false-positive results, it makes the size of the indices very small and independent from the security parameter at the cost of leaking number of matches.

Boneh et al. introduced the first Public-key Encryption with Keyword Search (PEKS) [17], which was shown later that it leaks user's access pattern. Furthermore, [17] is insecure against offline keyword guessing attack [18]. Boneh et al. proposed another PEKS [19], which is based on PIR [20] and Bloom filters, where the aim is to hide the access pattern. Although [19] is secure, PIR-based schemes are computationally expensive. Moreover, in [19] the number of matches that can be found in the remote storage is fixed beforehand to not leak the number of matches. To reduce the search overhead, Bellare et al. [21] introduced an efficient public-key searchable encryption (ESE), which achieved an optimal search time. In contrast to PEKS, ESE allows other users to generate tokens and search for data in the remote storage unit only by having the public key. However, ESE encryption scheme is deterministic and vulnerable to brute-force attacks.

Sahai and Waters [22] introduced a new encryption scheme called Attribute-Based Encryption (ABE), which is capable of using an arbitrary string as the public key. In ABE, a ciphertext is not generated for a particular receiver, but for whom possess the desired attributes. In later works, Goyal et al. [23] and Bethencourt et al. [5] revised the ABE and introduced Key-Policy Attribute-Based Encryption (KP-ABE) and Ciphertext-Policy Attribute-Based Encryption (CP-ABE). Then, Han et al. [24] proposed a new encryption scheme, Attribute-Based Encryption with Keyword Search (ABEKS), which enables a multi-user access control based on KP-ABE. In ABEKS, a token is generated by using user's private key, and it consists of the desired data to be searched in the remote storage unit. However, in [24] search is realized through decryption of ciphertexts in the database, which introduces a significant computational overhead. Moreover, this technique requires generating a trapdoor, which necessitates collaboration with data owner. [25] proposes a privacy-preserving ranked multi-keyword search in a multi-owner model. This approach allows each data owner to use his private key for the encryption. However, this proposal suffers from the high computational cost of searching. Guo et al. [26] proposed a secure search that supports multiple data owners setting. [26] also enables rank search based on the relevance of documents and keyword, and quality of documents. They also propose an efficient indexing structure, group keyword balanced binary tree (GBB tree), to achieve higher efficiency in searching. However, in [26], a trusted third proxy is used to facilitate data outsourcing and query evaluation. Moreover, the improvement over BB-tree leaks private information regarding access pattern. As it is stated in [26], the GBB-tree may not access one or multiple subtrees to re-

duce computation overhead, which can reveal access pattern.

Chung et al. [6] introduced a secure outsourcing protocol based on Gentry's fully homomorphic encryption scheme [27]. Although in [27] confidentiality of data is preserved, while data processing remains possible, its computational overhead is still a challenge. Li et al. [28] proposed a method to apply homomorphic encryption with an overhead linear in the number of the records. Xiong et al. [29] introduced a ciphertext-policy-ABE (CP-ABE) searchable encryption by using homomorphic encryption, where its search time is proportional to the size of the dataset. Bösch et al. [30] proposed a scheme, BTH$^+$, which is a combination of somewhat homomorphic encryption, and indexing technique of Chang and Mitzenmacher [11]. In Bösch's work [30], only the data owner can perform a search over the data, since generating trapdoor requires the possession of the private key.

Gentry et al. [31] proposed a secure searching based on ORAM and Somewhat Homomorphic Encryption (SHE) scheme. Although using ORAM in [31] prevent information leakage in searching, it is communication-wise expensive. Popa et al. [32] introduced one of the most well-known solutions, CryptDB, for searching over encrypted data. CryptDB uses different encryption schemes depending on the type of the given query to be evaluated over encrypted data. However, CryptDB leaks the number of matches to the untrusted server. Moreover, in [33], the authors show that CryptDB is insecure because it does not provide integrity for the query. Krell et al [34] introduce another secure searching using ORAM, SHE, and Bloom filter that is significantly more efficient. However, they achieve such efficiency at the cost of leaking access pattern. Moreover, the work in [34] suffers from high storage complexity, where 100K of records each having four searchable keywords results in an encrypted index that using 75GB of RAM. Another drawback of [34] is that it requires the data owner to be online for searching. Table IV.1.1 summarizes the performance and security of the state-of-the-art searching techniques and our secure searching protocols. In Table IV.1.1, we denote multi-reader and multi-writer setting as M-M, n is the total number of records, n_v^a is the number of records having the attribute a equal to v, d is the number of data owners, λ is a security parameter, and k stands for top-k documents as described in [26]. Note that, in contrast to the existing works, our proposal only addresses the problem of finding the matching records, not retrieving them. Thus, the current works and ours are not comparable concerning performance, communication/computation-wise.

1.3. SECURE SEARCHING PROTOCOLS

In this work, there are five parties: 1) users, 2) data storage units, 3) query issuer, 4) remote computation system, and 5) key manager:

1. Key Manager (KM): KM generates a pair of public and private keys and shares the public key with the other parties. KM also collaborates with RCS to perform two-party computations such as secure decryption.

2. Users: They are the owners of private data (patients) that are stored in remote data storage in encrypted form. The users send their measurements to the corresponding vendors. The data is consisting of several attributes, denoted by p_i. Therefore, the users' data structure is a tuple $T(id_u, p_i, id_{p_i}, v)$, where id_u is the user identity,

Table IV.1.1: Summary of schemes. Communication round is denoted as CR, data transmission as DT, and statistical query as SQ.

Scheme	Search	CR	DT	Leakage	SQ	"AND"	M-M
[8]	$\mathcal{O}(n\log^2 n)$	$\mathcal{O}(\log n)$	$\mathcal{O}(\log^2 n)$	no leakage	no	no	no
[34]	$\mathcal{O}(\log n)$	$\mathcal{O}(\log^2 n)$	$\mathcal{O}(\log^3 n)$	access pattern	no	yes	no
[31]	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$	$\mathcal{O}(n\log^2 n)$	no leakage	no	yes	no
[26]	$\mathcal{O}(n)$	$\mathcal{O}(d)$	$\mathcal{O}(k)$	access pattern	no	yes	yes
[16]	$\mathcal{O}(\log 2^m + n_v)$	$\mathcal{O}(1)$	$\mathcal{O}(n_v)$	access pattern	no	no	no
[14]	$\mathcal{O}(n_v)$	$\mathcal{O}(1)$	$\mathcal{O}(\lambda)$	access pattern	no	no	no
IBSvI	$\mathcal{O}(mn)$	$\mathcal{O}(1)$	$\mathcal{O}(n)$	no leakage	yes	yes	yes
IBSvII	$\mathcal{O}(mn)$	$\mathcal{O}(1)$	$\mathcal{O}(n)$	no leakage	yes	yes	yes

p_i is an attribute (e.g., date, time, age, device id, etc), id_{p_i} is an unique identity for p_i, and v is the measurement. The users send their encrypted tuples to their data storage units.

3. Data Storage Unit (DSU): Each data storage unit collects data from one or multiple users and sends the data to a remote computation system, cloud, on a regular basis. In our scenario, DSUs are the vendors, who are offering smart devices to people.

4. Query Issuer (QI): QI is interested in processing users' data (i.e., hospital or medical research institutes). In our work, QI can ask for generating statistics like counting, averaging, and data aggregating. QI constructs a query that includes one or multiple attributes as $q : \{Q_T, \acute{p}_i, \acute{id}_{p_i}\}$ based on the type of result that QI is interested in. Q_T defines the type of the query like counting, each $\acute{p}_i$ represents the value of an attribute. To prevent private data leakage, QI encrypts p_i values. id_{p_i} are meta-data that describe $\acute{p}_i$ referring to what type of attribute (i.e., blood pressure or heart beating rate).

5. Remote Computation System (RCS): RCS is a considered to be a cloud storage and processing unit that has sufficient computational and storage capacity. RCS receives and stores the encrypted data from all DSUs. RCS also receives encrypted queries from QI. Henceforward, we denote the jth tuple in RCS as T_j.

For our constructions, we use an additively homomorphic and semantically secure encryption scheme, namely Paillier [35], and a fully homomorphic scheme, Fan-Vercanteren (FV) [36]. Our system is designed under the assumption of semi-honest security model [37]. In our setting, the aim is to protect private data of users and QI from both KM and RCS, which are semi-trusted, during the evaluation of the query q provided by QI. The results of queries are kept hidden from KM and RCS.

1.3.1. IBSvI

Searching in IBSvI consists of two phases: First, we generate indices, which is done before the execution of the protocol (off-line phase) and second, we invoke the searching

Table IV.1.2: List of symbols

Symbol	Description	Symbol	Description
pk/sk	public/secret key	n	encryption modulus
$\mathscr{E}_{pk}$	encryption	$\mathscr{D}_{sk}$	decryption
κ	security parameter	r, r_z, θ	random number in $\mathscr{Z}_n$
T	tuple of multiple attributes	p	attribute
α	number of attributes in a query	$bl(p)$	bit length of p
ℓ	maximum bit-length of attributes	φ	false positive rate control
Ans	result of query evaluation	$\acute{\rho}$	package capacity in Paillier
Q_T	query type	id_{p_i}	meta data for p_i
UnPerm	inverse of Perm	$[x]$	$\mathscr{E}_{pk}(x)$
$\hat{\rho}$	number of item can be packed in FV	index^{p_i}	index tables for p_i
$\text{index}^{p_i}_{i,j}$	i^{th} column and j^{th} row in index^{p_i}	$d(a,b)$	Hamming distance of a and b
$T^{p_i}_j$	value of p_i in j^{th} record	$(T^{p_i}_j)_i$	i^{th} least significant bit of $T^{p_i}_j$
$(p_i)_j$	j^{th} least significant bit of p_i	ω	total number of records
Perm	permutation function	Max_α	total number of attributes

protocol upon receiving the query q from QI (on-line phase).

GENERATING INDICES

Indices are constructed in four steps:

1. Users change the measurements of each attribute p_z, $z \in \{0, \cdots, \alpha - 1\}$ to binary form $(p_z)_i$, where $i \in \{0, \cdots, e \leq \ell - 1\}$ and e is the bit-length of p_z. Then, they assign zero to the rest of bits from $(p_z)_e$ to $(p_z)_{\ell-1}$.

2. Users multiply each $(p_z)_i$ by 2^i, $i \in \{0, \cdots, e\}$, encrypts the results as a tuple $T^{p_z} :< id_{p_z}, [id_u], [(p_z)_i] >$ and send them to their DSUs, who send the encryptions to RCS later.

3. RCS creates an index for each possible attribute (there are Max_α attributes in total), where each index has ω rows that is the total number of tuples collected from DSUs and ℓ columns.

4. RCS locates each encrypted tuple $[(T^{p_z})_{i,j}]$ (i^{th} bit of the jth tuple received from DSUs) in the corresponding generated index, $index^{[p_z]}_{i,j} \leftarrow [(T^{p_z})_{i,j}]$, $i \in \{0, \cdots, \ell - 1\}$. $index^{[p_z]}$ refers to the index that contains encrypted integers of attribute p_z.

We present values in the binary form to check whether each encrypted data stored by RCS matches a particular encrypted data in q without invoking any two-party protocol. Each value is multiplied by powers of 2 to eliminate false positive results in our construction. Protocol 10 shows the steps that RCS takes to generate indices. It indicates that Max_α indices are created, one for each attribute. According to Protocol 10, it is explicit generating indices in RCS is computation-free and storage-demanding.

Protocol 10 RCS:GenIndex

Input: $[T_j^{p_z}]$
Output: indices for each p_z

```
for z = 0 to Max_α − 1 do
    Create index^{p_z}
    for i = 0 to ℓ − 1 do
        for j = 0 to ω − 1 do
            index_{i,j}^{[p_z]} ← [(T^{p_z})_{i,j}]
        end for
    end for
end for
```

SECURE SEARCHING

Protocol 2 shows the process of evaluating an encrypted query $[q]$ over the encrypted databases using the generated indices.

1. $QI_{f_1}^I$: Once RCS generates the indices, QI constructs a query q including $e \leq \alpha$ attributes $\acute{p}_z$, $z \in \{0, \cdots, e-1\}$, which QI is interested in. Similar to the process of index generation, QI converts the values of its attributes to binary form. Then it computes $(\acute{p}_z)_i \leftarrow (\acute{p}_z)_i * (-2^i)$, $i \in \{0, \cdots, e-1\}$ and assigns zero to $\{(\acute{p}_z)_e, \cdots, (\acute{p}_z)_{\ell-1}\}$.

2. $RCS_{f_1}^I$: RCS tosses a random coin r for each pair of $(\text{index}_{i,j}^{p_z}, (\acute{p}_z)_i)$ to compute $(\text{index}_{i,j}^{p_z} \oplus (r * 2^i), (\acute{p}_z)_i \oplus (-r * 2^i))$. Note that the values of $\text{index}_{i,j}^{p_z}$ is in encrypted form. Thus, to compute $\text{index}_{i,j}^{p_z} \oplus (r * 2^i)$, RCS checks whether $r = 0$, $\text{index}_{i,j}^{[p_z]} \leftarrow (r = 0)?(\text{index}_{i,j}^{[p_z]}) : ([2^i] - \text{index}_{i,j}^{[p_z]})$. Similarly, to compute $(\acute{p}_z)_i \oplus (-r * 2^i)$, RCS performs $[(\acute{p}_z)_i] \leftarrow (r = 0)?[(\acute{p}_z)_i] : (-([2^i] + [\acute{p}_z)_i])$. Afterwards, RCS computes $[d_{i,j}^{p_z}]$, which is clearly zero if $|\text{index}_{i,j}^{p_z}| = |(\acute{p}_z)_i|$ that means the ith bit of $\text{index}_j^{p_z}$ matches the ith bit of $\acute{p}_z$ in q.

3. $RCS_{f_2}^I$: To check if $\acute{p}_z$ equals p_z of the jth tuple in the database, RCS checks whether the Hamming distance d between them is zero, $d(\text{index}_j^{p_z}, \acute{p}_z) = 0$. To obtain d, RCS computes $R_j^{p_z} \leftarrow \sum_{i=0}^{\ell-1} [d_{i,j}^{p_z}]$.

4. $RCS_{f_3}^I$: In case of there are multiple $\acute{p}_z$ in query q, RCS checks if the all the $\acute{p}_z$ matches the corresponding $[p_z]$ in each tuple. To do so, RCS can simply compute $[y_j] \leftarrow \sum_{z=0}^{\alpha-1} [R_j^{p_z}]$, if $[y_j = 0]$ then it means the jth tuple in RCS matches $[q]$. However, there is a possibility of obtaining false positive result. Thus, RCS chooses α uniformly distributed random number r_z, $z \in \{0, \cdots, \alpha-1\}$ and another random number θ. r_z values are used in the protocol to decrease the false positive rate and θ is used for the security reasons.

5. $RCS_{f_4}^I$: In this step RCS choose ω uniformly random numbers $\acute{y}_j \in \{0, \cdots, 2^{\ell-1}\}$ and another random number $\acute{r}$. In this step, RCS inserts ω random numbers, which has

Protocol 2: IBSvI

Require: index^{p_z}

Ensure: Find matches

$e \in \{0, \cdots, u \leq \alpha - 1\}$ $\quad QI^I_{f_1}$

$[q] \leftarrow \{Q_T, [-2^i(p'_e)_i], id_{p'_e}\}$

for $z = 0$ **to** $\alpha - 1$ **do**

  for $i = 0$ **to** $\ell - 1$ **do**

    for $j = 0$ **to** $\omega - 1$ **do**

      $r \xleftarrow{\$} \{0, 1\}$ $\quad RCS^I_{f_1}$

      $\text{index}^{p_z}_{i,j} \leftarrow \text{index}^{p_z}_{i,j} \oplus (r * 2^i)$

      $(p'_z)_i \leftarrow (p'_z)_i \oplus (-r * 2^i)$

      $[d^{p_z}_{i,j}] \leftarrow (\text{index}^{p_z}_{i,j} + [(p'_z)_i])$

    end for

  end for

end for

for $j = 0$ **to** $\omega - 1$ **do**

  for $z = 0$ **to** $\alpha - 1$ **do**

    $[R^{p_z}_j] \leftarrow [0]$

    for $i = 0$ **to** $\ell - 1$ **do**

      $[R^{p_z}_j] \leftarrow [R^{p_z}_j + d^{p_z}_{i,j}]$ $\quad RCS^I_{f_2}$

    end for

  end for

  $r_z \xleftarrow{\$} \{2^{\varphi-1}, \cdots, 2^{\varphi} - 1\}$ $\quad RCS^I_{f_3}$

  $\theta \xleftarrow{\$} \{1, \cdots, Max_\alpha - \alpha + 1\}$

  $[y_j] \leftarrow [\sum_{z=0}^{\alpha-1}(R^{p_z}_j * \theta r_z)]$

end for

$\acute{y}_j \xleftarrow{\$} \{0, \cdots, 2^{\ell - \lceil}\}$ $\quad RCS^I_{f_4}$

$\acute{r} \xleftarrow{\$} \{0, \cdots, \omega - 1\}$

$\text{InsertZero}(\acute{y}, \acute{r})$

$[y] \leftarrow [y] || [\acute{y}]$

$\text{Perm}([y_j]), j \in \{0, \cdots, 2\omega - 2\}$ $\quad RCS^I_{f_5}$

$y_j \leftarrow Dec_{key}([y_j])$ $\quad KM^I_{f_1}$

$I_j \leftarrow (y_j = 0)?1 : 0$

$[I_j] \leftarrow Enc(Ans_j)$

$[I_j] \leftarrow \text{Reverse-Perm}([I_j])$ $\quad RCS^I_{f_6}$

$[Ans_j] \leftarrow \text{Half}([I_j]) : j \in \{0, \cdots, \omega - 1\}$

$\acute{r}$ zeros, to $[y]$. This step prevent the KM to learn about the number of matches and other statistical information from y_j values.

6. $RCS^I_{f_5}$: RCS permutes $[y_j]$ to not let the KM learn about the locations of tuples that matched q.

7. $KM^I_{f_1}$: The KM decrypts given $[y_j]$, $j \in \{0, \cdots, 2\omega - 2\}$, and checks whether $y_j = 0$. Then, it creates an array I_j, filled with binary values such that $I_j \leftarrow (y_j = 0)?1 : 0$.

8. $RCS^I_{f_6}$: RCS reverse permutes $[I_j]$ and removes the dummy encryptions added to $[y_j]$ in RCS_{f_4}.

Finally, RCS has an encrypted binary array, which represents the location of matching tuples in the database. To optimize IBSvI regarding computation and communication, we also apply data packing [38, 39] on our protocol. Note that we cannot apply

data packing on $[I_j]$ because it prevents performing reverse permutation and removing dummy encryptions in $RCS^I_{f_6}$. However, we can skip $RCS^I_{f_6}$, and perform $RCS^I_{f_4}$ and $RCS^I_{f_5}$ on the database itself, where $\acute{j} = \acute{r} = 0$, for evaluation of Q_T. This modification allows to pack $[I_j]$ and reduces both communication and computation cost.

1.3.2. IBSvII

In IBSvII, we achieve a communication cost-free searching algorithm. Similar to IBSvI, there are two phases in IBSvII, indexing and searching. The index tables generation phase is identical to IBSvI except we do not multiply $(p_z)_i$ by 2^i.

Protocol 3 shows how IBSvII works. There are four steps in IBSvII, $QI^{II}_{f_1}$, $RCS^{II}_{f_1}$, $RCS^{II}_{f_2}$, and $RCS^{II}_{f_3}$.

Protocol 3: IBSvII

Require: index^{p_z}

Ensure: Find matches

$e \in \{0, \cdots, u \leq \alpha - 1\}$ $\quad QI^{II}_{f_1}$
$[q] \leftarrow \{Q_T, [-(\acute{p}_e)_i], id_{\acute{p}_e}\}$
for $z = 0$ **to** $\alpha - 1$ **do**
  for $i = 0$ **to** $\ell - 1$ **do**
    for $j = 0$ **to** $\omega - 1$ **do**
      $[d^{p_z}_{i,j}] \leftarrow (\text{index}^{p_z}_{i,j} + [(\acute{p}_z)_i])^2$ $\quad RCS^{II}_{f_1}$
    end for
  end for
end for
for $j = 0$ **to** $\omega - 1$ **do**
  for $z = 0$ **to** $\alpha - 1$ **do**
    $[R^{p_z}_j] \leftarrow [1]$
    for $i = 0$ **to** $\ell - 1$ **do**
      $[R^{p_z}_j] \leftarrow [R^{p_z}_j \times d^{p_z}_{i,j}]$ $\quad RCS^{II}_{f_2}$
    end for
  end for
  $[Ans_j] \leftarrow [\prod_{z=0}^{\alpha-1} R^{p_z}_j]$ $\quad RCS^{II}_{f_3}$
end for

1. $QI^{II}_{f_1}$: Similar to $QI^I_{f_1}$, QI generates the query with $e \leq \alpha$ attributes p_z, $z \in \{0, \cdots e - 1\}$. Then, QI represents the the values of the attributes to binary form, encrypts them, and send $\{[1-(p_z)_0], \cdots, [1-(p_z)_{\ell-1}]\}$ to RCS. Afterward, QI sends Q_T plus necessary meta-data to RCS.

2. $RCS^{II}_{f_1}$: This step computes $[d^{p_z}_{i,j}]$ like in $RCS^I_{f_1}$, however, $[d^{p_z}_{i,j}]$ are binary values. The challenge in computing $[d^{p_z}_{i,j}]$ in $RCS^I_{f_1}$ was that it could be a combination of positive and negative numbers such that their addition $R^{p_z}_j$ becomes zero, which is considered as a false positive result. In $RCS^{II}_{f_1}$, each $[d^{p_z}_{i,j}]$ is squared, which solves the problem of negative numbers. Recall that $(\text{index}^{p_z}_{i,j} + (\acute{p}_z)_i) \in \{-1, 0, 1\}$, thus $d^{p_z}_{i,j} \in \{0, 1\}$ in $RCS^{II}_{f_1}$.

3. $RCS^{II}_{f_2}$: After computation of $[d^{p_z}_{i,j}]$, we compute $R^{p_z}_j \leftarrow R^{p_z}_j \times d^{p_z}_{i,j}$. $R^{p_z}_j$ remains one if the $\acute{p}_z$ in the query q matches p_z of jth record in the database.

4. $RCS^{II}_{f_3}$: This step apply the "AND" connections among multiple $R^{p_z}_j$ by computing $\prod_{z=0}^{\alpha-1} R^{p_z}_j$ to obtain Ans_j.

In IBSvII, we can use batching to reduce the computational overhead. Batching enables not only the addition of two packed ciphertexts but also supports multiplication. Thus, all the operations stated in Protocol 3 can be performed over packed ciphertexts without collaboration with KM.

1.4. SECURITY ANALYSES

We consider the semi-honest security model [37], where parties are assumed to be honest in following the protocol description, while they are curious to obtain more information than they are entitled to. Given that the only RCS gets is the encrypted output from the protocol, KM should not be able to distinguish if RCS has a different input and RCS should not learn more information than the output of the protocol. We also assume that parties do not collude with each other.

1.4.1. SECURITY OF IBSVI

Let $RCS^I_f = (RCS^I_{f_1}, \cdots, RCS^I_{f_6})$, $KM^I_f = (KM^I_{f_1})$, and $f = (RCS^I_f, KM^I_f)$ to be the PPT functionality for IBSvI. The view of the ith party ($i \in \{RCS, KM\}$) during the execution of IBSvI on $(index^{p_z}, \phi)$ and security parameter n is denoted by $view^{IBSvI}_i(index^{p_z}, \phi, n) = (w, r^i; m^i_1, \cdots, m^i_t)$, where $w \in \{index^{p_z}, \phi\}$ based on the values of i, r^i are the ith party internal random numbers, and m^i_j represents the jth message that is received by ith party. Note that KM does not have any initial input, thus its input is denoted as ϕ. $output^{IBSvI}_i(index^{p_z}, \phi, n)$ represents the output of each party during the execution of IBSvI. To represent the joint output of both parties, we denote

$$\begin{aligned} output^{IBSvI} = (&output^{IBSvI}_1(index^{p_z}, \phi, n), \\ &output^{IBSvI}_2(index^{p_z}, \phi, n)). \end{aligned} \tag{1.1}$$

Definition 1.4.1. It can be proven that IBSvI securely computes $f = (RCS^I_f, KM^I_f)$ in the semi-honest security setting if there exits PPT algorithms Sim_{RCS} and Sim_{KM} such that:

$$\begin{aligned} \{(Sim_{RCS}(1^n, index^{p_z}, RCS^I_f, f))\} &\overset{c}{\equiv} \{(view^f_{RCS} \\ (index^{p_z}, \phi, n), output^f(index^{p_z}, \phi, n))\} \end{aligned} \tag{1.2}$$

and

$$\begin{aligned} \{(Sim_{KM}(1^n, \phi, KM^I_f, f))\} &\overset{c}{\equiv} \{(view^f_{KM} \\ (index^{p_z}, \phi, n), output^f(index^{p_z}, \phi, n))\} \end{aligned} \tag{1.3}$$

Theorem 8. *The protocol IBSvI is simulation secure and securely computes the functionality f, when the party RCS is corrupted by adversary $\mathscr{A}_{RCS}$ in the presence of semi-honest adversaries.*

Proof. We need to show that RCS cannot computationally distinguish between generated messages and outputs from $\mathscr{S}_2$ and $\mathscr{S}_3$, and randomly generated data. RCS receives an output from $\mathscr{S}_2$, $[\hat{Ans}_j]$, and a message from $\mathscr{S}_3$, $[(\hat{p}_z)_i]$. Protocol 6 shows the simulators for KM and QI that are $\mathscr{S}_2$ and $\mathscr{S}_3$, respectively. Note that because of the space limitation, only the message from $\mathscr{S}_3$ to RCS, $[\hat{p}_z)_i]$, is presented in Protocol 6. Given $index^{p_z}$ and 1^n (security parameters), RCS works as follow:

1. RCS chooses three uniformly random tapes r, r_z, and $\acute{r}$ for RCS_f.
2. $\mathscr{S}_3$ randomly generates u $\hat{p}_z \in \mathbb{N}$, where $u \leq \alpha$, u random meta data, and a random $\hat{Q}_T$. Then, $\mathscr{S}_2$ forms $[q] \leftarrow \{\hat{Q}_T, [(\hat{p}_z)_i] || \hat{MD}_i\}$, where $i \in \{0, \cdots, u \leq \alpha - 1\}$, ans send $[q]$ to RCS.
3. RCS executes $RCS^I_{f_i}, i \in \{1, \cdots, 5\}$, and it outputs $[\hat{y}_j]$ to $\mathscr{S}_2$.
4. $\mathscr{S}_2$ tosses j coins $\hat{I}_j$ and sends $[\hat{I}_j]$ to RCS.
5. RCS performs UnPerm$[\hat{I}_j]$, and outputs $[\hat{Ans}_j] \leftarrow$Half$([\hat{I}_j])$

The output of the simulation can be written as:

$$Sim_{RCS}(1^n, index^{p_z}, RCS^I_f, f) = (index^{p_z}, r, r_z, \acute{r}; [\hat{I}_j], [(\hat{p}_z)_i]; ([\hat{Ans}_j], \phi)].$$

The real view of RCS can be presented as

$$view^f_{RCS}(index^{p_z}, [(\acute{p}_z)_i, n]) = (index^{p_z}, r, r_z, \acute{r}; [I_j], [(\acute{p}_z)_i])$$

. And the output of the real view is $output^f(index^{p_z}, [(\acute{p}_z])_i) = ([Ans_j], \phi)$. It can be observed that the encryption pairs $([\hat{I}_j], [(\hat{p}_z)_i]))$ and $([I_j], [(\acute{p}_z)_i])$ are indistinguishable, since the crypto-scheme used in IBSvI is semantically secure. For the same reason $\mathscr{A}_{RCS}$ cannot distinguish between $[Ans_j]$ and $[\hat{Ans}_j]$. Recalling that RCS is also given meta-data that describes the query type and attributes in q, RCS cannot see if the provided meta-data are corresponding to the attributes $\hat{p}_z$ in q. Therefore, we can claim that

$$\begin{aligned} Sim_{RCS}(1^n, index^{p_z}, RCS^I_f, f) \overset{c}{\equiv} \{view^f_{RCS} \\ (index^{p_z}, [(\acute{p}_z)_i], n), output^f(index^{p_z}, [(\acute{p}_z)_i]). \end{aligned} \tag{1.4}$$

□

Theorem 9. *The protocol IBSvI is simulation secure and securely computes the functionality f, when the party KM is corrupted by $\mathscr{A}_{KM}$ in the presence of semi-honest adversaries.*

Proof. The simulation for the case when KM is corrupted is presented in Protocol 7. After receiving 1^n, KM works as follows:

1. $\mathscr{S}_1$ chooses a uniformly random number $\hat{r} \in \{0, \cdots, 2\omega - 2\}$.

2. $\mathscr{S}_1$ chooses 2ω uniformly random numbers $\hat{y}_j \in \{-v, \cdots, v\}$, where $v = Max_\alpha(2^\ell - 1)(2^\phi - 1)$, that contains $\hat{r}$ zeros.

3. $\mathscr{S}_1$ encrypts $\hat{y}_j$ and sends the permuted encryptions $[\hat{y}_j]$ to KM.

4. KM calls the KM_{f_1} functionality to obtain encryptions $[I_j]$ and send them to $\mathscr{S}_1$.

5. $\mathscr{S}_1$ performs UnPerm$[I_j]$, and outputs $[Ans_j] \leftarrow$Half$([I_j])$.

The simulation and the real view can be written as:

$$Sim_{KM}(1^n, \phi, KM_f, f) = (\phi; [\hat{y}_j]; [Ans_j]]). \tag{1.5}$$

The view and output of KM are

$$view^f_{KM}(index^{p_z}, \phi, n) = (\phi, [y_j])$$

and

$$output^f(index^{p_z}, [(\acute{p}_z)_i]) = (\phi, [Ans_j]).$$

Since $\mathscr{A}_{KM}$ has the decryption key, we need to show that $\mathscr{A}_{KM}$ cannot distinguish between y_j and $\hat{y}_j$. We need to consider following points to prove the security theorem:

1. The values of $(index^{p_z})_i$ and $(p_z)_i$: as it is presented in Protocol 2, both indices $(index^{p_z})_i$ and $(p_z)_i$ are XORed with uniformly distributed random r in $RCS^I_{f_1}$.

2. The bit-lengths of $(index^{p_z})_i$ and $(p_z)_i$: to hide the bit-lengths of $(index^{p_z})_i$ and $(p_z)_i$, a fix bit-length solution is suggested, where $(index^{p_z})_i$ and $(p_z)_i$ are ℓ bits for every entry.

3. Number of attributes α in q: the value of $alpha$ has a direct effect on the upper and lower bounds of y_j. To prevent $\mathscr{A}_{KM}$ to learn about the α, as it is shown in $RCS^I_{f_3}$, RCS multiplies $[y]_j$ by the difference between α and the maximum number of attributes that QI can put in q, Max_α.

4. Number of zeros in y_j: by learning number of zeros from multiple y_j, $\mathscr{A}_{KM}$ might be able to distinguish between real y_j and $\hat{y}_j$. $RCS^I_{f_4}$ randomizes the number of zeros by inserting a random number of zeros to y_j.

Randomizing the stated four properties guarantees that $\mathscr{A}_{KM}$ cannot distinguish between y_j and $\hat{y}_j$, thus:

$$\begin{aligned} Sim_{KM}(1^n, \phi, KM_f, f) = \{view^f_{KM} \\ (index^{p_z}, \phi, n), output^f(index^{p_z}, [(\acute{p}_z)_i])\}. \end{aligned} \tag{1.6}$$

□

1.4.2. SECURITY OF IBSVII

In IBSvII, computation of Ans_j does not require collaboration of RCS and KM; thus, RCS can compute the final binary vector Ans_j without any communication. To prove the security of Protocol 3, we need to show that RCS cannot learn anything from data.

Theorem 10. *The protocol IBSvII is simulation secure and securely computes the functionality f, when the party RCS is corrupted by adversary $\mathscr{A}_{RCS}$ in the presence of semi-honest adversaries.*

Proof. We need to show that RCS is unable to computationally distinguish between the truly generated messages given from $\mathscr{S}_3$, the simulator of QI, and randomly generated data. Given $index^{p_z}$ and 1^n (security parameters), RCS in IBSvII works as follows:

1. $\mathscr{S}_3$ randomly generates u $\hat{p}_z \in \mathbb{N}$, where $u \leq \alpha$, u random meta data, and a random $\hat{Q}_T$. Then, $\mathscr{S}_2$ forms $[q] \leftarrow \{\hat{Q}_T, [(\hat{p}_z)_i] || \hat{MD}_i\}$, where $i \in \{0, \cdots, u \leq \alpha - 1\}$, ans send $[q]$ to RCS.

2. RCS executes $RCS^I_{f_i}, i \in \{1, \cdots, 3\}$, and then outputs $[Ans_j]$.

The output of the simulation can be presented as:

$$Sim_{RCS}(1^n, index^{p_z}, RCS^{II}_f, f) = (index^{p_z}, [(\hat{p}_z)_i]; [\hat{Ans}_j], \phi).$$

The real view of RCS is $view^f_{RCS}(index^{p_z}, [(\acute{p}_z)_i]) = (index^{p_z}, [(\acute{p}_z)_i])$ and the real view of the output is $output^f(index^{p_z}, [(\acute{p}_z)_i]) = ([Ans_j], \phi)$. Clearly, the encryptions $[(\hat{p}_z)_i]$ and $[(\acute{p}_z)_i])$, and $[Ans_j]$ and $[\hat{Ans}_j]$ are indistinguishable because of using semantically secure crypto-scheme. Meta-data in q also does not reveal any information about the attributes. Therefor, we can claim that

$$\begin{aligned} Sim_{RCS}(1^n, index^{p_z}, RCS^I_f, f) \overset{c}{\equiv} \{view^f_{RCS} \\ (index^{p_z}, [(\acute{p}_z)_i]), output^f(index^{p_z}, [(\acute{p}_z)_i]). \end{aligned} \tag{1.7}$$

□

1.5. PERFORMANCE ANALYSES

1.5.1. COMPLEXITY ANALYSIS

Table IV.1.3 shows the computational complexity of our protocols for searching with multiple attributes (IBSvI and IBSvII). In Table IV.1.3, we present x number of exponentiations with y-bit exponents as $(x)_y$. As it is illustrated in Table IV.1.3, the complexity of IBSvI in terms of the number of additions is linear to the number of attributes in q, bit-length of attributes, and the number of records in the database. Moreover, Table IV.1.3 shows how applying data packing reduces the number of homomorphic additions and decryptions. Table IV.1.3 also shows IBSvII does not require any encryption and decryption; however, it requires performing $(2\alpha\omega\ell + \alpha\omega)/\hat{\rho}$ homomorphic multiplications, where $\hat{\rho}$ is the number of messages that can be packed by deploying batching.

Table IV.1.3: Computational complexity of the searching protocols. Addition is denoted as Add, exponentiation as Expo, multiplication as Mult, encryption as Enc, and decryption as Dec.

Protocols	Add	Expon	Mult	Enc	Dec.
IBSvI	$\frac{3\alpha\omega\ell+\alpha\omega}{\acute{\rho}}$	$\frac{\alpha\omega_{\log(\theta r_z)}}{\acute{\rho}}$	0	3ω	$\frac{2\omega}{\acute{\rho}}$
	$\mathcal{O}(\frac{\alpha\omega\ell}{\acute{\rho}})$		—	$\mathcal{O}(\omega)$	$\mathcal{O}(\frac{\omega}{\acute{\rho}})$
IBSvII	$\frac{\alpha\omega\ell}{\hat{\rho}}$	0	$\frac{2\alpha\omega\ell+\alpha\omega}{\hat{\rho}}$	0	0
	$\mathcal{O}(\frac{\alpha\omega\ell}{\hat{\rho}})$		$\mathcal{O}(\frac{\alpha\omega\ell}{\hat{\rho}})$	—	—

Table IV.1.4 summarizes the communicational complexity of the searching protocols in terms of data transmission, communication round, and storage complexity. According to Table IV.1.4, communication round for IBSvI is constant, and it is independent of the bit-length of inputs and the number of attributes in q. Moreover, Table IV.1.4 shows that RCS in IBSvII computes Ans_j without any communication with KM. Furthermore, data transmission needed is independent of the number of AND conjunctions used in the query.

Table IV.1.4: Communicational and space complexity of the searching protocols. Data transmission complexity is denoted as Data.Trans, communication round as Com.Rond, and index size as IS.

Protocols	Data.Trans	Com.Rond	IS
IBSvI	$4\omega/\acute{\rho}$	1	$\omega\ell\alpha/\acute{\rho}$
IBSvII	0	0	$\omega\ell\alpha$

Table IV.1.4 shows the data transmission needed for the secure searching protocols per party, RCS, and KM, where the complexity of IBSvI is $\mathcal{O}(\omega)$ and independent of the number of attributes. Recall that there is no communication between RCS and KM in IBSvII, and clearly, no data is transmitted between these two parties. The batching technique significantly reduces computational costs. Table IV.1.4 shows the size of each protocol's index table. The value of $\acute{\rho}$ depends of bit-length of ℓ, where smaller ℓ results in larger $\acute{\rho}$.

1.5.2. EXPERIMENTAL RESULTS

In this section, we present the experimental results of implementing IBSvI and IBSvII. First, we show the run-times of the introduced searching protocols for different α and φ values. Then, we compare their run-times in RCS and KM for different α values. To obtain the run-times of the protocols, we use C++ and external libraries: MPIR, Boost, the Secure Computation Library (SeComLib), and SEAL on a single Linux machine running Ubuntu 14.04 LTS, with a 64-bit microprocessor and 8 GB of RAM. We applied a simple parallelization technique in our implementation (4 threads). The cryptographic key length of the Paillier is chosen according to NIST standards [40], which are valid until

2030. Table V.1.4 lists the parameters and their values in our implementation.

Table IV.1.6 compares the run-times of the IBSvI, IBSvII, and existing works in RCS and KM. To show the trade-off between computational and communicational costs when using different encryption schemes, we also provide the run-time of IBSvI when FV is used, denoted as IBSvI_{FV}. Note that Paillier cannot be used in IBSvII because it does not support the ciphertext multiplications. Table IV.1.6 points out both $\text{IBSvI}_{\text{Paillier}}$ and IBSvI_{FV} demand significantly low computational resources from RCS. These results show that Paillier equipped with data packing and FV with enabled batching have the same performance in performing homomorphic addition. However, there is a noticeable difference between the run-times of $\text{IBSvI}_{\text{Paillier}}$ and IBSvI_{FV} in RCS, which means the decryption function in FV is more efficient than Paillier because of being able to pack more messages into a single ciphertext, $\hat{\rho} > \acute{\rho}$. Table IV.1.6 presents the run-time of IBSvII that is roughly similar to the total run-time of $\text{IBSvI}_{\text{Paillier}}$. Considering these results, we can conclude that, contrary to popular belief, using fully homomorphic crypto-schemes is more efficient than partially homomorphic ones in certain cases.

Table IV.1.6 shows the run-time of secure searching in [34], where only the cost of finding the matching records in that work is considered. According to Table IV.1.6, the work in [34] is also efficient in finding the desired records at the costs of leaking access pattern with an index table size of 75 Gigabytes. Note that the network cost of searching is not provided separately in [34]; therefore, we put the total network cost (searching and retrieving) in Table IV.1.6.

Table IV.1.5: Parameters and their values.

Parameter	Symbol	Value
Bit-size of inputs	ℓ	15 bits
Number of records in RCS	ω	10^5
Security parameter	κ	112 bits
Number of queries to obtain FPR		10^7
Capacity of a package (IBSvI)	$\acute{\rho}$	89
Capacity of a package (IBSvII)	$\hat{\rho}$	2048

Table IV.1.6: Run-times and communication costs of the IBSvI and IBSvII. For [34], we use NA to show that the work does not report the corresponding value.

Protocols	Run-time (second)		Data transmission	index size	communication
	RCS	KM	(megabyte)	(megabyte)	(round)
$\text{IBSvI}_{\text{Paillier}}$	0.525	5.06	2.19	8.2	1
IBSvI_{FV}	0.44	0.45	49.45	92.8	1
$\text{IBSvII}_{\text{FV}}$	5.54	0	0	92.8	0
[34]	4.0		26	75K	NA

1.6. CONCLUSION

Searching in encrypted databases is a challenging task due to the complexity introduced by encryption. In this work, we focus on a medical setting where institutions would like to use the data collected by smart wearables from several vendors to analyze them for the well-being of the patients. In order to make the system usable in practice, we propose a two-step procedure: 1) creation of index tables at the time of uploading data from vendors to the cloud storage unit, and 2) executing queries using these tables. Our complexity analysis and experimental results on a large dataset clearly show the contribution of our work: the performance of the system is outstanding. It is also worth mentioning that our idea of creating index tables can be generalized to other application settings, introducing a scalable and efficient search mechanism for encrypted databases where data will later be processed under encryption.

REFERENCES

[1] R. W. Treskes, L. A. van Winden, N. van Keulen, D. E. Atsma, E. T. van der Velde, E. van den Akker-van Marle, B. Mertens, and M. J. Schalij, *Using smart technology to improve outcomes in myocardial infarction patients: Rationale and design of a protocol for a randomized controlled trial, the box,* JMIR research protocols **6** (2017).

[2] P. G. Goldschmidt, *HIT and MIS: implications of health information technology and medical information systems,* Commun. ACM **48**, 68 (2005).

[3] M. Meingast, T. Roosta, and S. Sastry, *Security and privacy issues with health care information technology,* in *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE* (IEEE, 2006) pp. 5453–5458.

[4] M. E. Johnson, *Data hemorrhages in the health-care sector,* in *Financial Cryptography and Data Security, 13th International Conference, FC 2009, Accra Beach, Barbados, February 23-26, 2009. Revised Selected Papers* (2009) pp. 71–89.

[5] J. Bethencourt, A. Sahai, and B. Waters, *Ciphertext-policy attribute-based encryption,* in *2007 IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May 2007, Oakland, California, USA* (2007) pp. 321–334.

[6] K. Chung, Y. T. Kalai, and S. P. Vadhan, *Improved delegation of computation using fully homomorphic encryption,* in *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings* (2010) pp. 483–501.

[7] R. Ostrovsky, *Efficient computation on oblivious rams,* in *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA* (1990) pp. 514–523.

[8] O. Goldreich and R. Ostrovsky, *Software protection and simulation on oblivious rams,* J. ACM **43**, 431 (1996).

IV.1

[9] D. X. Song, D. 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.

[10] E. Goh, *Secure indexes,* IACR Cryptology ePrint Archive **2003**, 216 (2003).

[11] Y. Chang and M. Mitzenmacher, *Privacy preserving keyword searches on remote encrypted data,* in *Applied Cryptography and Network Security, Third International Conference, ACNS 2005, New York, NY, USA, June 7-10, 2005, Proceedings* (2005) pp. 442–455.

[12] E. Stefanov, C. Papamanthou, and E. Shi, *Practical dynamic searchable encryption with small leakage,* in *21st Annual Network and Distributed System Security Symposium, NDSS 2014, San Diego, California, USA, February 23-26, 2014* (2014).

[13] R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky, *Searchable symmetric encryption: Improved definitions and efficient constructions,* Journal of Computer Security **19**, 895 (2011).

[14] S. Kamara, C. Papamanthou, and T. Roeder, *Dynamic searchable symmetric encryption,* in *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012* (2012) pp. 965–976.

[15] X. Ding, P. Liu, and H. Jin, *Privacy-preserving multi-keyword top-k similarity search over encrypted data,* IEEE Transactions on Dependable and Secure Computing **PP**, 1 (2017).

[16] R. Miyoshi, H. Yamamoto, H. Fujiwara, and T. Miyazaki, *Practical and secure searchable symmetric encryption with a small index,* in *Secure IT Systems - 22nd Nordic Conference, NordSec 2017, Tartu, Estonia, November 8-10, 2017, Proceedings* (2017) pp. 53–69.

[17] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, *Public key encryption with keyword search,* in *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings* (2004) pp. 506–522.

[18] J. W. Byun, H. S. Rhee, H. Park, and D. H. Lee, *Off-line keyword guessing attacks on recent keyword search schemes over encrypted data,* in *Secure Data Management, Third VLDB Workshop, SDM 2006, Seoul, Korea, September 10-11, 2006, Proceedings* (2006) pp. 75–83.

[19] D. Boneh, E. Kushilevitz, R. Ostrovsky, and W. E. S. III, *Public key encryption that allows PIR queries,* in *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings* (2007) pp. 50–67.

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

[21] M. Bellare, A. Boldyreva, and A. O'Neill, *Deterministic and efficiently searchable encryption,* in *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings* (2007) pp. 535–552.

[22] A. Sahai and B. Waters, *Fuzzy identity-based encryption,* in *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings* (2005) pp. 457–473.

[23] 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, Ioctober 30 - November 3, 2006* (2006) pp. 89–98.

[24] F. Han, J. Qin, H. Zhao, and J. Hu, *A general transformation from KP-ABE to searchable encryption,* Future Generation Comp. Syst. **30**, 107 (2014).

[25] W. Zhang, Y. Lin, S. Xiao, J. Wu, and S. Zhou, *Privacy preserving ranked multi-keyword search for multiple data owners in cloud computing,* IEEE Trans. Computers **65**, 1566 (2016).

[26] Z. Guo, H. Zhang, C. Sun, Q. Wen, and W. Li, *Secure multi-keyword ranked search over encrypted cloud data for multiple data owners,* Journal of Systems and Software (2017).

[27] 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.

[28] M. Li, J. Li, and C. Huang, *A credible cloud storage platform based on homomorphic encryption,* Netinfo Security **12**, 35G40 (2012).

[29] A.-P. Xiong, Q.-X. Gan, X.-X. He, and Q. Zhao, *A searchable encryption of cp-abe scheme in cloud storage,* in *Wavelet Active Media Technology and Information Processing (ICCWAMTIP), 2013 10th International Computer Conference on* (IEEE, 2013) pp. 345–349.

[30] C. Bösch, Q. Tang, P. H. Hartel, and W. Jonker, *Selective document retrieval from encrypted database,* in *Information Security - 15th International Conference, ISC 2012, Passau, Germany, September 19-21, 2012. Proceedings* (2012) pp. 224–241.

[31] C. Gentry, S. Halevi, C. S. Jutla, and M. Raykova, *Private database access with he-over-oram architecture,* in *Applied Cryptography and Network Security - 13th International Conference, ACNS 2015, New York, NY, USA, June 2-5, 2015, Revised Selected Papers* (2015) pp. 172–191.

[32] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, *Cryptdb: processing queries on an encrypted database,* Commun. ACM **55**, 103 (2012).

IV.1

[33] I. H. Akin and B. Sunar, *On the difficulty of securing web applications using cryptdb,* IACR Cryptology ePrint Archive **2015**, 82 (2015).

[34] F. Krell, G. Ciocarlie, A. Gehani, and M. Raykova, *Low-leakage secure search for boolean expressions,* in *Topics in Cryptology - CT-RSA 2017 - The Cryptographers' Track at the RSA Conference 2017, San Francisco, CA, USA, February 14-17, 2017, Proceedings* (2017) pp. 397–413.

[35] 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.

[36] J. Fan and F. Vercauteren, *Somewhat practical fully homomorphic encryption,* IACR Cryptology ePrint Archive **2012**, 144 (2012).

[37] O. Goldreich, *The Foundations of Cryptography - Volume 2, Basic Applications* (Cambridge University Press, 2004).

[38] 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).

[39] J. R. Troncoso-Pastoriza, S. Katzenbeisser, M. U. Celik, and A. N. Lemma, *A secure multidimensional point inclusion protocol,* in *Proceedings of the 9th workshop on Multimedia & Security, MM&Sec 2007, Dallas, Texas, USA, September 20-21, 2007* (2007) pp. 109–120.

[40] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, *Nist sp800-57: Recommendation for key management part 1: General (revised),* NIST, Tech. Rep (2007).

IV.2 | Knapsack Based Data Packing

Abstract

Processing encrypted data is a well-known solution when protecting privacy-sensitive data from untrusted processing units. However, data expansion, as a result of data encryption, makes undesired computational and communicational overheads in the cryptographic applications. Data packing is one of the useful tools to minimize the overheads. In this work, we introduce a novel approach for packing encrypted data based on the subset sum problem. We show that our data packing achieve high performance in reducing the overheads and it is significantly more efficient than existing techniques. Moreover, we show that our approach perfectly matches with secure searching protocols for secure data retrieval.

This chapter has been published as "A Novel Approach For Data Packing: Using Trapdoor Knapsack", by M.Nateghizad, Z.Erkin, and R.L.Lagendijk in the proceedings of 10th *International Workshop on Information Forensics and Security*, 2018.

IV.2

2.1. INTRODUCTION

Data Packing (DP) techniques [1–4] are widely used in cryptographic applications [5–8] to reduce communication and computation costs. DP enables storing multiple private values in a single plaintext to use the message space of homomorphic crypto-systems efficiently. Moreover, a set of encrypted values packed in a ciphertext can be treated as a single encrypted value when performing operations such as addition or inner product [4] between two packages. This property is exploited in many works to reduce the number of cryptographic operations [6] and enhance the performance of decryption [9].

Existing techniques of DP are based on concatenation that can be simply realized. Assume $[a]_i$ are the encryptions of ℓ-bit integer values, and n is the bit-length of the message space of a crypto-system. The number of $[a]_i$ can be packed into one package is $\rho = \lfloor n/\ell \rfloor$, and the packing process is as follows [2]:

$$[\hat{a}] = [\sum_{i=0}^{\rho-1} a_i (2^\ell)^i] = \prod_{i=0}^{\rho-1} [a_i]^{(2^\ell)^i}. \tag{2.1}$$

As it is shown in Equation 2.1, the package $[\hat{a}]$ is divided into ρ pieces, where each piece can hold similar or different integer values. The addition of two packages $[\hat{a}]$ and $[\hat{b}]$ yields a new package such that $[\hat{c}] = [\hat{a} + \hat{b}] = \prod_{i=0}^{\rho-1} [a_i + b_i]^{(2^\ell)^i}$.

There are several limitations and challenges in the existing DPs: 1) the concatenation-based DP (CDP) splits the total space of the message space into fixed-size pieces, since the exact bit-lengths of each encrypted integers are not publicly known for security reason, 2) changing a single value in a package is computationally expensive since it requires building another package with similar construction, and 3) encrypted zero takes as much space as an encrypted non-zero integer in a package. The last argument is important since the response of a search query on encrypted databases result in an encrypted vector such that the majority of the values in the vector are encrypted zeros accompanied with a few encrypted non-zero that match the query [10]. Since it is impossible to distinguish between encrypted zero and non-zero values, many packages should be created to include all the ciphertexts, where a few packages could be enough for the encrypted non-zero values.

In this work, we introduce a novel approach for packing encrypted data that is based on the Subset Sum Problem (SSP) [11]. SSP is a well-known NP-complete problem [12], where given a set $A = \{a_i : 1 \leq i \leq n\}$ of integers and another positive integer M, the problem is finding a subset of A has sum equal to M [11]. Many crypto-schemes constructed based on unique SSP (uSSP) [13–16]. To use SSP in a crypto-scheme, there should be only one unique subset of A associated with one M that is called uSSP; otherwise, the decryption process may not give the correct result.

We exploit a property of uSSPs that allows to efficiently find the associated subset of a sum M using trapdoor to build a new data packing. Unlike the existing DPs that relies on concatenation, our approach, uSSP based DP (uSSP-DP), packs encrypted data by only performing addition. uSSP-DP overcomes difficulties in CDP: 1) it can insert encrypted values into a package such that they take minimum possible spaces without leaking any information, 2) changing one of the values in a package is a matter of one addition, and more importantly 3) encrypted zero values do not occupy any space when

they are inserted into a package. We also show that uSSP-DP requires less computational resources to build a package than CDP. Note that we only use SSP for packing encrypted data and we rely on other crypto-systems for data encryption, since using SSP to encrypt data can be insecure [17].

To show how uSSP-DP can be beneficial, we make a realistic scenario in the medical domain, and we show that uSSP-DP outperforms CDP regarding communication and computation costs. As it is shown in Figure IV.2.1, there are three parties in our scenario: 1) remote storage and computation system, 2) medical institute, and 3) key manager. Remote Storage and Computation System (RCS) has sufficient computational and storage capacity. RCS receives and stores the measurements (blood pressure, blood sugar, heart rate) from all patients and their IDs in encrypted form to prevent violation of medical data [18]. RCS also receives encrypted queries from MI and evaluates the query with the help of the key manager. Medical Institute (MI) is interested in searching, retrieving, and processing patients' data stored in the RCS's database. QI encrypts his query before sending it to RCS to prevent information leakage. Key Manager (KM) generates a pair of public and private keys and shares the public key with the other parties. KM also collaborates with RCS and MI to perform two-party computations such as secure searching and decryption.

The system in Figure IV.2.1 has two main phases: 1) data searching (step 3) and 2) data retrieving (steps 7 and 8). In step 7, RCS sends the generated encrypted vector in step 6 to RI. To save the communication cost in steps 7 and 8, and the computation cost in step 8 [9], CDP can be used to pack the encrypted vector in step 6. As one of the limitations of CDP, it cannot efficiently pack a set of ciphertexts when most of them are encrypted zero after performing secure search. This limitation causes a significant increase in the communication cost in steps 7 and 8, and the computation cost in step 8. In this work, we show that using uSSP-DP instead of CDP can significantly decrease computational and communicational costs in steps 7 and 8 by reducing the number of the package through increasing the number of encrypted non-zero values in each package.

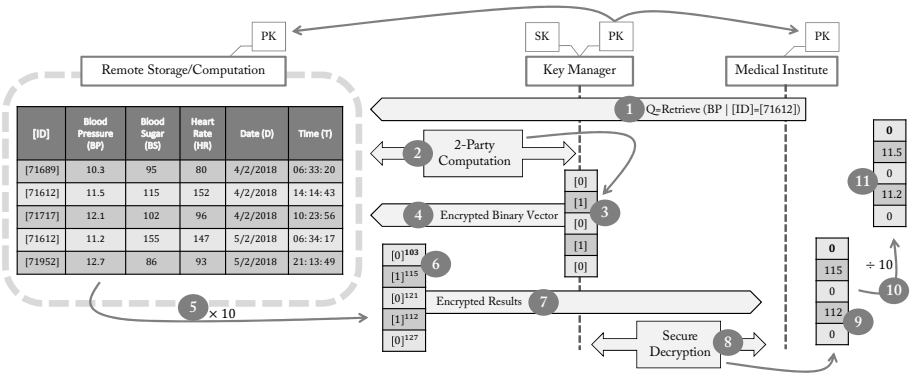

Figure IV.2.1: Secure Data Searching and Retrieval

2.2. PRELIMINARIES

2.2.1. SUBSET SUM PROBLEM AND ITS VARIATIONS

In this section, we explain different variations of SSP-based crypto-schemes. MerKle and Hellman [11] introduced two trapdoor knapsacks based crypto-system: 1) by using Super Increasing Subset Sum problem (SISS) and 2) Multiplicative Trapdoor Knapsack (MTK). In SISS based crypto-system, first two large numbers m and w are chosen such that $w < m$ and $gcd(m, w) = 1$. Second, a super increasing knapsack vector is chosen $\acute{a} = (\acute{a}_1, \acute{a}_2, \cdots, \acute{a}_n)$ such that $\acute{a}_i > \sum_{j=1}^{i-1} \acute{a}_j > 0$. Next step is to build a trapdoor knapsack vector by computing $a_i = \acute{a}_i \cdot w \bmod m$ to obtain $a = (a_1, a_2, \cdots, a_n)$. To encrypt a binary vector $x = (x_1, x_2, \cdots, x_z)$, where $z \leq n$, the value $\mathscr{S} = \sum_{i=1}^{z} a_i \cdot x_i$ is computed. Decryption of $\mathscr{S}$ and obtaining the binary vector x are achieved as follows: 1) we compute $\acute{\mathscr{S}} = \mathscr{S} \cdot w^{-1} \bmod m$ is computed, 2) if $\acute{a}_n \leq \acute{\mathscr{S}}$, then $x_n = 1$; otherwise, $x_n = 0$, 3) for x_i, $2 \leq i \leq n$, $x_i = 0$ if and only if $\acute{a}_i \leq \acute{\mathscr{S}} - \sum_{j=i+1}^{n} x_j \cdot \acute{a}_j$.

In MTK, a set of co-primes number are chosen $\acute{a} = (\acute{a}_1, \acute{a}_2, \cdots, \acute{a}_n)$, a large number m such that $gcd(\acute{a}_i, m) = 1$ and $\prod_{i=1}^{n} \acute{a}_i \leq m$, and a based b that is co-prime to m and $\acute{a}_i$. To generate the knapsack trapdoor, the vector $a = (a_1, a_2, \cdots, a_n)$ is computed such that $b^{a_i} = \acute{a}_i \bmod m$, which is a discrete logarithm problem but it can be solved easily if the logarithms are taken over $GF(m)$ [19]. The encryption of a message that is represented as a binary vector x is obtained by computing $\mathscr{S} = \sum_{i=1}^{z-1} a_i \cdot x_i$. To decrypt $\mathscr{S}$ and obtain the vector x, first $\acute{\mathscr{S}} = b^{\mathscr{S}} \bmod m$ is computed, and then by checking if any of $\acute{a}_i$ divides $\acute{\mathscr{S}}$, it is possible to reconstruct the vector x. Although, obtaining the vector a in MTK requires solving discrete logarithms, but unlike SISS, MTK supports encrypting a set of integer numbers, where the numbers can be repeated in the set several times.

SSP-based encryption mentioned above are only capable of encrypting a binary vector x. The Compact Knapsack problem (CK) [20] enables encryption of a vector x containing decimal positive integers. To construct the crypto-system in [20], first, a positive integer $w = 2^b - 1$ is chosen, where b is an positive integer. Second, a set of positive integers in the range of $[0, w]$ as the domain D is formed. Third, the parameter b is chosen based on the fact the a message m is partitioned into several pieces of nb-bit, where n is the number of pieces and b is the bit-length of each piece. Then, n pairs of (q_i, k_i) are created such that all the q_i's are co-prime to each other, the conditions $w \leq k_i$ and $q_i \bmod k_i \neq 0$ hold for each i, and $k_i w \leq q_i \bmod k_i$. Afterwards, $R_i = q_i \bmod k_i$ and P_i values are computed such that $P_i \bmod q_i = R_i$ and $P_j \bmod q_i = 0$ if $i \neq j$. Compute $N_i = \lceil q_i/(k_i \cdot P_i) \rceil$ and $s_i = P_i \cdot N_i \bmod Q$, where $Q = \prod_{i=1}^{n} q_i$.

For a partitioned message m, the ciphertext is computed as $C = m \cdot s = \sum_{i=1}^{n} m_i \cdot s_i$. The partitioned message can be recovered from C by computing $m_i = \lfloor k_i \cdot C/q_i \rfloor \bmod k_i$. There are more different variations of crypto-systems based on knapsack problem; however, to the best of our knowledge, recent works focused on improving either the security or the performance of the existing crypto-systems, and the basis of their construction that are based on the SSP are not changed significantly.

2.2.2. ADDITIVE HOMOMORPHIC ENCRYPTION

In an additive homomorphic encryption scheme such as [21], multiplying two ciphertexts $\mathscr{E}_{pk}(m_1)$ and $\mathscr{E}_{pk}(m_2)$ results in a ciphertext, whose decryption is the sum of two

plaintexts m_1 and m_2: $\mathscr{D}_{sk}(\mathscr{E}_{pk}(m_1)\cdot\mathscr{E}_{pk}(m_2)) = (m_1+m_2) \bmod n$, where n is the encryption system modulus. Consequently, exponentiation of any ciphertext with a public integer value k yields the encrypted product of the original plaintext and the public value: $\mathscr{D}_{sk}(\mathscr{E}_{pk}(m)^k) = (k\cdot m) \bmod n$. For the encryption and decryption operations in Paillier, we refer readers to [21]. In the rest of this paper, we use $[x]$ to show the encryption of x.

2.3. DATA PACKING BASED ON KNAPSACK PROBLEMS

2.3.1. DATA PACKING USING SISS

Let $M = (msg_1, msg_2, \cdots, msg_z)$, where each $msg_j \in \{0,\cdots,n\}$ be the messages to be encrypted and outsourced in a remote storage. The process of the data packing using trapdoor knapsack is described in Protocol 11. In [11], SISS was used as an encryption scheme; thus it is important to generate $\mathscr{S}$ such that it does not leak any information about the values in the package. To provide such security, Merkle and Hellman [11], multiplicatively mask each value in vector $\acute{a}$ by using w, and then unmask $\mathscr{S}$ at the end by using multiplicative inverse of w, $w^{-1} \bmod m$. However, we are not using SISS for encrypting data, but only for packing. Therefore, we can directly use the non-mask form of vector $\acute{a}$ for packing, which saves computation and increases the ρ value.

Protocol 11 Data (un)packing using SISS

1: Generates a super increasing vector $\acute{a} = (\acute{a}_1, \acute{a}_2, \cdots, \acute{a}_n)$ such that n is the total number of values that a message msg can take.
2: Each message msg_j in M is encoded to $\acute{msg}_j = \acute{a}_{msg_j}$ for each $j \in \{1,\cdots,z\}$, $\acute{M} = (\acute{msg}_1, \cdots, \acute{msg}_z)$.
3: The encryption version of each message $\acute{msg}_j$ is computed, $[\acute{msg}_j] = E_{Paillier}(\acute{msg}_j)$.
4: Any set of t encrypted messages, $t \le z$, $[\acute{msg}_j]$ can be packed into one ciphertext by performing addition $[\mathscr{S}] = [\sum_{j=1}^{t} \acute{msg}_j] = \prod_{j=1}^{t} [\acute{msg}_j]$.
5: To unpack a decrypted package $\mathscr{S}$ to the binary vector $x = (x_1, x_2, \cdots, x_n)$, first check if $\acute{a}_n \le \mathscr{S}$, then $x_n = 1$. Then, for the rest of x_i values, $i \in \{n-1, n-2, \cdots, 1\}$, if and only if $\acute{a}_i \le \mathscr{S} - \sum_{j=i+1}^{n} x_j \cdot \acute{a}_j$, then $x_i = 1$; otherwise, $x_i = 0$.
6: Each i, where $x_i = 1$, is one of the msg_j.

After initializing the knapsack vectors, the process of packing continues with replacing msg_i in M by the corresponding $\acute{a}_{msg_i}$ to obtain $\acute{M}$. Since $\mathscr{S}$ is the addition of a set of $\acute{a}_i$ values, it is possible to recover the packed messages, vector x, such that i values where $x_i \neq 0$ are the messages msg_j. Note that having repeated inputs in SISS results in invalid unpacking. To solve this problem, we can simply combine uSSP-DPs and CDP techniques that is shown in Figure IV.2.2. First, we split the inputs into subsets of size z or less such that there is no similar values in each subset. Second, we pack each subset using SISS to obtain $[\mathscr{S}_i]$, and then we use CDP to put $[\mathscr{S}_i]$ in a package.

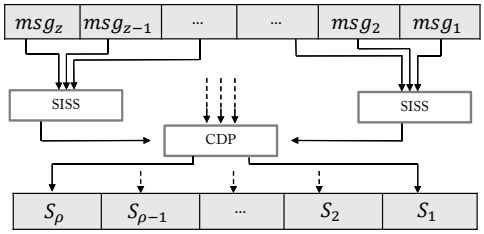

Figure IV.2.2: Using both SISS and CDP techniques.

2.3.2. DATA PACKING USING MTK PROBLEM

In both SISS and MTK, two knapsack vectors a and $\acute{a}$ are computed, where one vector is used for packing and another one for unpacking. The process of obtaining these two vectors in MTK is shown in Protocol 12. Recall that is using SISS in constructing a DP limits the inputs to be unique, unless we use both SISS and CDP in order to fully use the plaintext space in a package. Using MTK lifts that limitation of packing similar integers at the cost of more computation in the initialization phase. Assume that we have three inputs x_1, x_2, and x_3 where (x_1, x_2) are equal and x_3 is holding a different value. Consider a_1 and a_2 the corresponding values from the vector a for (x_1, x_2) and x_3, respectively. Note that since $x_1 = x_2$ they will be assigned the same value from vector a. Then, we can pack three inputs by computing $\mathscr{S} = a_1 + a_1 + a_2 = 2a_1 + a_2$. To unpack, first, $\acute{\mathscr{S}} = b^{2a_1+a_2} = \acute{a_1}^2 * \acute{a_2}$ is computed. Then, we check each $\acute{a}_i$ to see whether it divides $\acute{\mathscr{S}}$. Note that $\acute{a}_i$ values are co-primes, thus having similar values in a package is not problematic in unpacking phase.

Protocol 12 Data (un)packing using MTK

1: A set of n co-primes are generated, $\acute{a} = (\acute{a}_1, \acute{a}_2, \cdots, \acute{a}_n)$.
2: Choose m such that $gcd(\acute{a}_i, m) = 1$ and $\prod_{i=1}^{n} \acute{a}_i \leq m$.
3: Compute such b that is co-prime to $\acute{a}_i$ and m, then compute a_i such that $b^{a_i} = \acute{a}_i \bmod m$.
4: Each message msg_j in M is encoded to $\acute{msg}_j = a_{msg_j}$ for each $j \in \{0, 1, \cdots, z\}$, $\acute{M} = (\acute{msg}_1, \acute{msg}_2, \cdots, \acute{msg}_z)$.
5: The encryption version of each message $\acute{msg}_j$ is computed, $[\acute{msg}_j] = E_{Paillier}(\acute{msg}_j)$.
6: Any set of t encrypted messages, $t \leq z$, $[\acute{msg}_j]$ can be packed into one ciphertext by performing addition $[\mathscr{S}] = [\sum_{j=1}^{t} \acute{msg}_j] = \prod_{j=1}^{t} [\acute{msg}_j]$.
7: To unpack $\mathscr{S}$, and recover messages msg_i, first, $\acute{\mathscr{S}} = b^{\mathscr{S}} \bmod m$ is computed. Then, each $msg_j = i$ for $i \in \{1, 2, \cdots, n\}$, if and only if $\acute{a}_i$ divides $\acute{\mathscr{S}}$.

2.3.3. DATA PACKING USING CK PROBLEM

The process of packing encrypted data using compact knapsack problem is described in Protocol 13. Constructing a data packing using CK removes the need for encoding the

inputs; therefore, encrypted data can be packed directly. In both SISS and MTK, before packing, the inputs should be encoded to other values before encryption. Although data owners can encode the inputs to match the desired structure, encoding is not possible when only the encrypted form of data are available.

Protocol 13 Data (un)packing using CK

1: Choose b, and n according the number of inputs and their size.
2: Compute $w = 2^b - 1$, and generate a set of positive integers in the range of $[0, w]$ as the domain D.
3: Generate n pairs of (q_i, k_i), where $w \leq k_i$, $q_i \bmod k_i \neq 0$, and $k_i w \leq q_i \bmod k_i$.
4: Choose R_i and P_i such that $R_i = q_i \bmod k_i$, $P_i \bmod q_i = R_i$, and $P_j \bmod q_i = 0$, if $i \neq j$.
5: Compute $N_i = \lceil q_i/(k_i \cdot P_i) \rceil$ and $s_i = P_i \cdot N_i \bmod Q$, where $Q = \prod_{i=1}^{n} q_i$.
6: Packing the encrypted messages $[msg_i]$ is computed as $[\mathscr{S}] = [\sum_{i=1}^{n} msg_i \cdot s_i] = \prod_{i=1}^{n} [msg_i]^{s_i}$.
7: To recovery each individual message msg_i from the package $\mathscr{S}$, $msg_i = \lfloor k_i \cdot \mathscr{S}/q_i \rfloor \bmod k_i$ is computed for each $i \in \{1, \cdots, n\}$.

2.3.4. MODIFYING PACKAGES

Being able to modify a generated package $[\mathscr{S}]$ is a very useful property in many application. This modification includes adding a new value into a package or editing and removing an encrypted value in a package. Assume that we have two generated packages $\mathscr{S}_{uSSP-DP}$ and $\mathscr{S}_{CDP}$, then to add a new integer x to $[\mathscr{S}_{CDP}]$ we can simply follow Equation 2.1. First, we need to encrypt the value to get $[x]$. Then, we perform $[\mathscr{S}_{CDP}] = [2^\ell \mathscr{S}_{CDP} + x] = [\mathscr{S}_{CDP}]^{2^\ell} \cdot [x]$. Assume that $[\mathscr{S}_{CDP}] = [x_t || \cdots || x_p || \cdots || x_1]$ and each encrypted value takes ℓ bits space. In order to change $[x_p]$ to $[x_p + y]$, first, we compute $y2^{(p-1)\ell}$ and then encrypt it. Afterwards, by performing addition between $[y2^{(p-1)\ell}]$ and $[\mathscr{S}_{CDP}]$, we can add y to x_p in the package to obtain $[x_t || \cdots || x_p + y || \cdots || x_1]$. To remove an integer x from $[\mathscr{S}_{CDP}]$ (assume that $\mathscr{S}_{CDP}$ contains x), first, $[\mathscr{S}_{CDP}]$ should be decrypted, $\mathscr{S}_{CDP}$. Then, the package should be unpacked in order to find where x is located. After finding the location of x, it should be removed and the remaining values should be packed and encrypted again. It is clear that removing a value from $\mathscr{S}_{CDP}$ demands interaction between the party that holds the package and the key manager for decryption.

To add a new value x to $[\mathscr{S}_{uSSP-DP}]$, first, x should be encoded to $\acute{a}_x$ using the encoding vector, $\acute{a}$. Then, by performing $[\mathscr{S}_{uSSP-DP} + \acute{a}_x] = [\mathscr{S}_{uSSP-DP}] \cdot [\acute{a}_x]$, we can simply insert the encoded version of x into $[\mathscr{S}_{uSSP-DP}]$. Recall that SISS does not support packing similar values, thus adding a value in this packing technique is not recommended. Changing a value in a package that is constructed using uSSP, $[\mathscr{S}_{uSSP-DP}]$, is more challenging compared to CDP. Assume that SISS is used for data packing and $[\mathscr{S}_{SISS}] = [\acute{a}_{x_t} + \cdots + \acute{a}_{x_p} + \cdots + \acute{a}_{x_1}]$ and we want change $[\acute{a}_{x_p}]$ to $[\acute{a}_{x_p+y}]$. This modification can be easily done if the plaintext version of both x_p and y are available. Otherwise, it is not possible to change $[\acute{a}_{x_p}]$ in a SISS-DP structure, since we need to encode $x_p + y$ to $\acute{a}_{x_p+y}$. Using CK-DP instead of SISS-DP enables changing one of the values in

a package without having access to x_p and y in clear. The process of removing x from $[\mathscr{S}_{uSSP-DP}]$ is similar to adding x to $[\mathscr{S}_{uSSP-DP}]$. The only difference is that we perform $[\mathscr{S}_{uSSP-DP} - \acute{a}_x] = [\mathscr{S}_{uSSP-DP}] \cdot [\acute{a}_x]^{-1}$.

2.3.5. Linear Operations Over Packages

Another property of a packing technique that can be used in multi-party protocols to save computation costs is the ability to perform the linear operations over packages. This property is well-preserved in CDP, since it enables addition of two encrypted packages by simply performing one homomorphic addition, $[\mathscr{S}^1_{CDP} + \mathscr{S}^2_{CDP}] = [\mathscr{S}^1_{CDP}] \cdot [\mathscr{S}^2_{CDP}] = [\mathscr{S}^{1+2}_{CDP}]$. However, $\mathscr{S}^1_{CDP}$ and $\mathscr{S}^2_{CDP}$ should both have similar construction. If ℓ_1 and ℓ_2 are used to construct $\mathscr{S}^1_{CDP}$ and $\mathscr{S}^2_{CDP}$, respectively, based on Equation 2.1, and $\ell_1 \neq \ell_2$, then performing homomorphic addition between the two packages gives a wrong result. uSSP-DP also enables performing addition between two packages. However, if there are similar values in the two packages, then uSSP-DP techniques that support similarities such as MTK-DP should be used.

2.4. Performance Analysis

2.4.1. Complexity of Initialization

Although the three data packing techniques are based on the subset sum problem, their initialization phase takes a different amount of computational effort. In SISS, $\acute{a}_i$ are generated without computation, but only a condition that is $\acute{a}_i > \sum_{j=1}^{i-1} \acute{a}_j$. Thus, the data packing technique based on the SISS takes the least among of computational resources in the initialization phase. Data packing based on MTK requires solving discrete logarithm problems (DLP), which demands intensive computation. Although there are practical algorithms to efficiently solve DLP in special cases [19, 22], the complexity of MTK-based data packing is more than the two other data packing techniques. Choosing valid P_i and Q_i values in the compact knapsack, based on the two conditions stated in Protocol 13, can also be computationally expensive. An efficient solution, as it is suggested in [20], can be to choose $P_i = Q_i b_i$, where $Q_i = \prod_{i \neq j} q_i$ and $Q_i b_i = R_i$. Note that the parameters in the initialization phase in the three packing techniques need to be generated only once, and then can become publicly available for any application.

2.4.2. Complexity of Packing Encrypted Data

Unlike the initialization phase, the packing process is an online phase, and it may be repeated many times. As it is shown in Protocol 11 and Protocol 12, packing encrypted values into a package using SISS or MTK demands to perform only one homomorphic multiplication per value. However, as it is shown in Table IV.2.1, compact knapsack based data packing requires performing one exponentiation per value, which is computationally more expensive than multiplication. The complexity of a ciphertext modulo $\mathscr{N}$ with an x-bit exponent can be presented as $3x/2$ multiplications modulo $\mathscr{N}$ [23]. Note that using the compact knapsack, unlike SISS and MTK, we do not need to encode the original inputs to other values before packing. This advantage of compact knapsack can be beneficial where encoding the data before encryption is not feasible, and encrypted values should be packed directly at the cost of more computational complexity.

Table IV.2.1: Performance of packing protocols in terms of cost packing and the ρ value. ℓ is the bit-length of the encrypted values and $\mathcal{N}$ is the bit-length of message space.

Protocols	Packing ($\otimes$)	ρ
CDP	$\rho(3/2\ell+1)$	$\lfloor \mathcal{N}/\ell \rfloor$
SISS-DP	ρ	$\mathcal{N}-1$
MTK-DP	ρ	$\star$

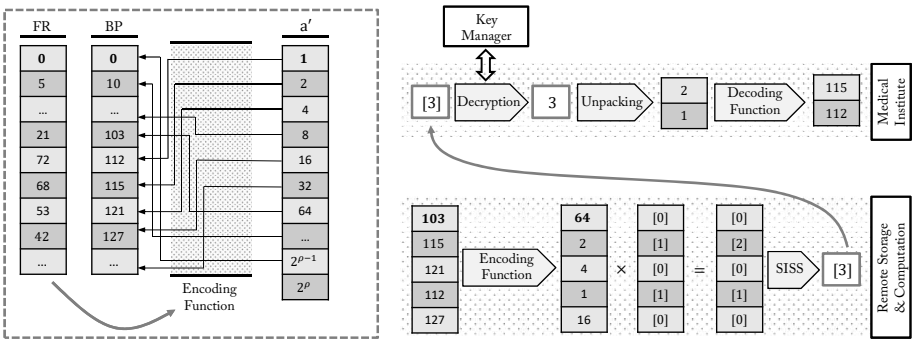

Figure IV.2.3: Encoding/Decoding process in SISS using optimized encoding vector.

2.4.3. PERFORMANCE IN TERMS OF ρ

One important property of a data packing technique is the number of values that they can pack into a package, ρ. The bit-length of messages ℓ and the message space of the underlying encryption scheme n are important factors in ρ. As it is presented in Table IV.2.1, in SISS, ρ is the size of knapsack vectors $\acute{a}$ that is equal to the bit-length of the message space $\mathcal{N}$. Computing ρ for the data packing based on MTK is more complicated. In MTK, the number of co-primes can be $n = \pi(\mathcal{N})$, where $\acute{a}_i \leq \mathcal{N}$. However, the a_i are distributed randomly in the rage of $[1, m]$, where $\mathcal{N} \ll \prod_{i=1}^{n} \acute{a}_i \leq m$. Therefore, n should be chosen with respect to the m; otherwise, some a_i may become larger than $\mathcal{N}$, which results in a faulty package.

To compare the performance of uSSP-DPs and CDP, we compute the ρ values for example in Figure IV.2.1 by using two different techniques SISS and concatenation. The ρ value when the concatenation is used in data packing is simple to compute, $\rho_{CDP} = \lfloor \mathcal{N}/\ell \rfloor$, where ℓ is the bit-length of the values to be packed. Let assume that $\mathcal{N} = 2048$ and $\ell = 8$, then $\rho_{CDP} = 256$. To compute ρ when SISS is used, ρ_{SISS}, first, we need to create the encoding vector. Since $\mathcal{N} = 2048$, minimum number of encrypted values that we can pack is 2047, $\rho_{SISS} = 2047$. Recall that in uSSP-DPs, packing encrypted zero does not take any space in the package. Moreover, to maximize the ρ_{SISS} value, the encoding vector should be created based on the ordered BP values by their frequency rates. As presented in Figure IV.2.3, encoding vector should be such that to encode the highest frequently BP to the lowest $\acute{a}_i$, second highest BP to the second lower $\acute{a}_j$, and so on.

Figure IV.2.3 shows that how the encoding vector can be ordered to maximize ρ value. Then, it shows how to use optimized SISS based DP to retrieve encrypted data as it is

shown in Figure IV.2.1. Table IV.2.2 shows more details about the packing techniques and their efficiencies. According to Table IV.2.2, the communication cost of sending 10^6 encryption of 5-bit integers without packing is 1 gigabyte, which is decreased to 2.4 megabytes and 1 kilobyte after using CDP and SISS-CDP, respectively. Interestingly, unlike the CDP, the communication cost of SISS-CDP remains unchanged for the larger inputs. That is because of two reasons: 1) the $\rho_{SISS-CDP} > 100$, 2) performance of SISS-CDP is independent of the bit-length of inputs because of the encoding process.

Table IV.2.2: Computation and communication costs of packing protocols, and steps 7 and 8 in Figure IV.2.1 for 10^6 encrypted integers, where less than 100 integers are non-zero. The size of plaintext and ciphertext in Pailler encryption scheme [21] are 2048 and 4096 bits, accordingly.

Bit-length	Packing cost ($\oplus$)		Steps 7 and 8 (Megabyte)			Step 8 (#decryption)		
	CDP	SISS-CDP	no packing	CDP	SISS-CDP	no packing	CDP	SISS-CDP
5	$8.3 * 10^6$	10^6	1024	2.4	$1.02 * 10^{-3}$	10^6	2444	1
10	$1.6 * 10^7$	10^6	1024	5.2	$1.02 * 10^{-3}$	10^6	4901	1
15	$2.3 * 10^7$	10^6	1024	7.4	$1.02 * 10^{-3}$	10^6	7352	1

Table IV.2.2 also shows the computation costs of packing 10^6 encrypted records. As it is shown in Table IV.2.2, the computation of data packing using CDP for 5-bit encrypted integers costs $8.3 * 10^6$ homomorphic additions ($\oplus$), which is 10^6 when SISS-CDP is used. Note that the cost of encrypted 10-bit integers using SISS-CDP is less than the cost for 5-bit encrypted integers. The reason is the fact that SISS is unable to unpack packages correctly when they contain similar integers. Thus, when the bit-length of inputs is 5-bit, there are 2^5 different integers to be packed; this means, although $\rho_{SISS} = 2047$, but we cannot pack more than 2^5 to avoid packing similar encrypted integers. In such a setting, there will be more SISS based packages $[\mathscr{S}_i]$ that are packed using CDP, Figure IV.2.2, which results in more computation cost and less overall ρ. Based on Table IV.2.2, using SISS-CDP to pack 10^6 encrypted integers also decreases the number of decryption (# decryption) in step 8, Figure IV.2.1, from 2444 to 1, which saves a significant amount of computation.

2.5. CONCLUSION

We proposed a novel approach for packing encrypted data based on the unique subset sum problem (uSSP). We introduced three Data Packing (DP) techniques based on uSSP that have different properties. We showed that uSSP-DP lifts the limitations and solves the problems of concatenation-based DP (CDP) in a realistic scenario and provided computational analysis to evaluate and compare the efficiencies of uSSP-DP and CDP. Due to the page restriction, we will extend this paper to include a security proof to show that our protocols are simulation secure in Arxiv.

REFERENCES

[1] J. R. Troncoso-Pastoriza, S. Katzenbeisser, M. U. Celik, and A. N. Lemma, *A secure multidimensional point inclusion protocol,* in *Proceedings of the 9th workshop on Multimedia & Security, MM&Sec 2007, Dallas, Texas, USA, September 20-21, 2007* (2007) pp. 109–120.

[2] 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).

[3] Z. Brakerski, C. Gentry, and S. Halevi, *Packed ciphertexts in lwe-based homomorphic encryption,* in *Public-Key Cryptography - PKC 2013 - 16th International Conference on Practice and Theory in Public-Key Cryptography, Nara, Japan, February 26 - March 1, 2013. Proceedings* (2013) pp. 1–13.

[4] M. Yasuda, T. Shimoyama, J. Kogure, K. Yokoyama, and T. Koshiba, *New packing method in somewhat homomorphic encryption and its applications,* Security and Communication Networks **8**, 2194 (2015).

[5] M. Barni, P. Failla, R. Lazzeretti, A. Sadeghi, and T. Schneider, *Privacy-preserving ECG classification with branching programs and neural networks,* IEEE Trans. Information Forensics and Security **6**, 452 (2011).

[6] Z. Erkin and G. Tsudik, *Private computation of spatial and temporal power consumption with smart meters,* in *Applied Cryptography and Network Security - 10th International Conference, ACNS 2012, Singapore, June 26-29, 2012. Proceedings* (2012) pp. 561–577.

[7] J. R. Troncoso-Pastoriza and F. Pérez-González, *Secure signal processing in the cloud: Enabling technologies for privacy-preserving multimedia cloud processing,* IEEE Signal Process. Mag. **30**, 29 (2013).

[8] R. L. Lagendijk, Z. Erkin, and M. Barni, *Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation,* IEEE Signal Process. Mag. **30**, 82 (2013).

[9] M. Nateghizad, Z. Erkin, and R. L. Lagendijk, *An efficient privacy-preserving comparison protocol in smart metering systems,* EURASIP J. Information Security **2016**, 11 (2016).

[10] M. Nateghizad, Z. Erkin, and R. L. Lagendijk, *Efficient index-based search protocols for encrypted databases,in press,* in *15th International Conference on Security and Cryptography (SECRYPT 2018).*

[11] R. C. Merkle and M. E. Hellman, *Hiding information and signatures in trapdoor knapsacks,* IEEE Trans. Information Theory **24**, 525 (1978).

[12] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (W. H. Freeman, 1979).

[13] A. Kate and I. Goldberg, *Generalizing cryptosystems based on the subset sum problem,* Int. J. Inf. Sec. **10**, 189 (2011).

[14] B. Wang, F. Li, and Y. Hu, *Improvement on a knapsack-based probabilistic encryption scheme,* IEICE Transactions **97-A**, 421 (2014).

[15] M. Kasahara, *A construction of A new class of knapsack-type public key cryptosystem, k(iii)sigma PKC,* IACR Cryptology ePrint Archive **2011**, 125 (2011).

[16] W. Zhang, B. Wang, and Y. Hu, *A new knapsack public-key cryptosystem,* in *Proceedings of the Fifth International Conference on Information Assurance and Security, IAS 2009, Xi'An, China, 18-20 August 2009* (2009) pp. 53–56.

[17] W. Baocang and Y. Hu, *Diophantine approximation attack on a fast public key cryptosystem,* in *Information Security Practice and Experience, Second International Conference, ISPEC 2006, Hangzhou, China, April 11-14, 2006, Proceedings* (2006) pp. 25–32.

[18] M. Meingast, T. Roosta, and S. Sastry, *Security and privacy issues with health care information technology,* in *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE* (IEEE, 2006) pp. 5453–5458.

[19] S. C. Pohlig and M. E. Hellman, *An improved algorithm for computing logarithms over gf(p) and its cryptographic significance (corresp.),* IEEE Trans. Information Theory **24**, 106 (1978).

[20] C. Lin, C. Chang, and R. C. T. Lee, *New public-key cipher system based upon the diophantine equations,* IEEE Trans. Computers **44**, 13 (1995).

[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] D. Coppersmith, *Fast evaluation of logarithms in fields of characteristic two,* IEEE Trans. Information Theory **30**, 587 (1984).

[23] T. Veugen, *Correction to "improving the DGK comparison protocol",* IACR Cryptology ePrint Archive **2018**, 1100 (2018).

V

Multiple Key Setting

V.1 | An Homomorphic Proxy Re-Encryption

Abstract

Homomorphic encryption has been widely used for developing cryptographic protocols that process privacy-sensitive data under encryption in scenarios where the data processing entity is not completely trusted. While it is possible to design such cryptographic protocols for any specific application, the research challenge has been on improving the efficiency of the protocols, namely computational overhead and communication cost. There have been many different application settings, where smart algorithms were proposed for better efficiency. However, very common approach in those works was to encrypt the data using the same key of a homomorphic encryption scheme. On the contrary, in a setting where there are several entities with different keys to encrypt their corresponding data, suggesting a cryptographic protocol for the realization of the specific algorithm under encryption is still a challenge. Indeed, there are Homomorphic Proxy Re-Encryption schemes (HPRES) which are capable of re-encrypting the ciphertexts to enable data processing under a different key but these schemes loose either homomorphism property or they are less efficient after re-encryption. In this work, we introduce a novel and more efficient HPRES that sustains homomorphism even after single/multiple time re-encryption. Furthermore, we also show that our HPRES is computationally more efficient than the state-of-the-art, up to 99%. Combined with data packing, our proposed scheme becomes more efficient to be used in cryptographic protocols as we demonstrate with a use case: data filtering within a large encrypted database that contains measurements from different personal health devices.

This chapter is under preparation to be submitted as "HOPE: A Homomorphic One-Direction Proxy Re-Encryption Scheme", by M.Nateghizad, Z.Erkin, and R.L.Lagendijk to 41th *IEEE Symposium on Security and Privacy*, 2020.

V.1

1.1. INTRODUCTION

Homomorphic encryption (HE) schemes are used to design cryptographic protocols that protect privacy-sensitive data by means of encryption while the encrypted data can still be processed without decryption. A wide range of applications from recommender systems [1] to biometric data matching [2] relies on the HE to achieve privacy-by-design solutions. Unlike other techniques such as data randomization and anonymization, HE keeps the original data unchanged and in its full form. Although using HE enables us to design privacy-preserving cryptographic protocols, there are a number of disadvantages: e.g. data expansion introduced by data encryption and computational overhead due to processing large numbers. Therefore, there is a significant amount of research on improving the building blocks of such cryptographic protocols such as equality tests[], comparison protocols [], distance computations [3, 4] and so on.

A common approach in privacy-preserving cryptographic protocols is to define an entity (Privacy Service Provider or Key Manager(KM)) which holds the decryption key. This entity then interacts with the others during the protocol. Unfortunately, having such an entity in real life is expensive and sometimes even impossible. On the contrary, most of the real world scenarios involve several entities which should process data jointly. In such scenarios, the challenge is then to share data with other entities and preferably data are encrypted using different keys and with homomorphism so that each entity can process them individually. An encryption scheme which preserves homomorphism even after re-encryption would result in less computation and communication overhead compared to the ones that suggest a KM.

Of course, a naive approach to share homomorphically encrypted data (HED) without KM can be proposed as follows: the data owner downloads all HED data, decrypts them, encrypts the clear data with another entity's public key, and sent the new encrypted data to that entity. Clearly, this naive approach is not practical: Consider a scenario where there are patients equipped with smart medical devices to capture a number of measurements such as weight, ECG, blood pressure, and blood sugar level []. These devices are usually connected to the smart phones of the patients, who need to be monitored on a daily basis. The smart phone application encrypts the measurements, as they are highly privacy-sensitive, with its corresponding public keys and sends the encrypted data to a remote storage unit (RS) which could be a cloud storage and processing unit that has sufficient computational and storage capacity in real life. RS receives and stores the encrypted data from all patients. There is another entity, e.g. a hospital, which is interested in processing patients' data in order to 1) generate statistics like counting and averaging and 2) perform targeted search within the encrypted measurements. Given the number of patients, the devices they might have and the frequency of the data produces, the amount of data can be in large orders []. Obviously, the naive approach is impractical.

As a better alternative, Homomorphic Proxy Re-Encryption schemes (HPRES) are a type of homomorphic encryption that enable data owners to share their already outsourced HED with others without the need for decryption. Although several HPRES are introduced [5–8], there are some concerns about their security or they are simply inefficient to use with large data sets. In this work, we introduced a Homomorphic One-direction Proxy-Encryption scheme (HOPE) which has the following properties:

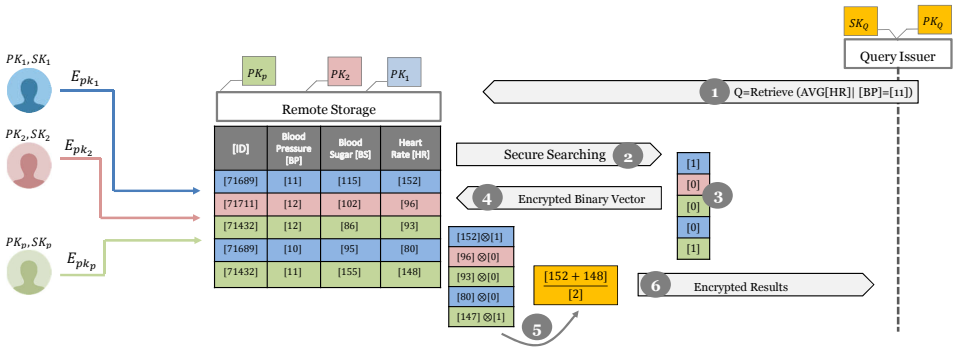

Figure V.1.1: Secure data filtering in a multiple-key setting

- Ciphertexts have homomorphism property before and after re-encryption.
- Re-encryption is one-direction (unidirectional).
- Re-encryption token does not leak private information.
- Re-encryption is completely performed in RS.
- The re-encryption key is generated in a non-interactive and collusion-free form [9].
- Re-encrypted ciphertexts can be re-encrypted again to be accessible for other parties, while they still have homomorphic property.
- Re-encryption process is extremely fast when it is compared with the state-of-the-art solutions.

To the best of our knowledge, HOPE is the first of its kind with all these properties in one scheme. Furthermore, when combined with data packing, which enables to encrypt multiple plaintexts in one ciphertext, HOPE becomes much more efficient to be used with large data sets. To prove our claim, we deploy HOPE on a secure search protocol from [10]. In the scenario, we assume that each patient is using a different key for encrypting his/her measurements. We then compare the performance of HOPE with the state-of-the-art HPRES.

In the rest of the paper, we first analyse the related works in Section II. Then, In Section III, we explain the preliminaries and tools used in the paper. In Section IV, we explain our protocol in detail. In Section V, we provide the analyses of our protocol with respect to security. Afterwards, we analyze the performance of our solution in Section VI. Finally, we conclude the paper in Section VII.

1.2. RELATED WORKS

Blaze et al. [11] introduced the notation of Proxy Re-Encryption (PRE), where they introduced a bidirectional PRE based on ElGamal encryption scheme. Their transformation

mechanism does not leak any private information to the proxy (the party that performs the transformation operation).

Ivan and Dodis in [12] introduced a simple generic proxy encryption and signature. They realized unidirectional proxy encryption for an IBE scheme, RSA, and ElGamal based on the idea of sharing the secret key. In fact, in [12], all the messages are encrypted with the global secret key in a proxy server, where each delegatee has access to the part of the key that can be used to transform the ciphertext to a particular form to be decrypted under a different key. Later Ayday et al. [13] combined a variant of homomorphic Paillier [14] encryption scheme and the work in [12] that resulted in a new homomorphic PRE scheme which is later used to build a privacy-preserving disease susceptibility test over patients' genomic data.

Although the PRE scheme in [12] is simple, there are three main drawbacks against using it. Fristly, for a new authorized delegate who wants to access a portion of data, all the ciphertexts that are already encrypted and stored should be updated. Secondly, colluding of proxy and delegatee reveal the global secret key that can be misused to decrypt all the encrypted messages. And thirdly, the proposed scheme is not secure as it is showed in [15].

Another unidirectional PRE with stronger security notions is introduced in [9], which relies on the decisional bilinear Diffie-Hellman (DBDH). Both [9, 12] are single-hop PRE, meaning that the construction does not allow to re-encrypt a ciphertext that is already re-encrypted once.

Later works aimed to investigate more into CCA2-secure PRE schemes [16–18]. Libert and Vergnaud [16] introduced the first unidirectional PRE with IND-CCA2 security without relying on random oracle assumption. Then, in [17, 18] more formal and developed security notions and attacks on PRE are provided to analyze the security of PRE schemes.

Xagawa [19] introduced the first approach in constructing a Learning with Error (LWE)-based PRE scheme. Then, [20, 21] addressed the security gap in [19]. Kirshanova [20] proposed a collusion-safe LWE-based PRE without using a trusted third party for generating re-encryption key. Chandran et al. [21] also present another PRE in their work, which is multi-hop and relies on decisional LWE assumption. Ma et al. [8] propose a single-hop secure homomorphic proxy re-encryption based on a fully homomorphic scheme in [22]. Although, the scheme in [5] supports homomorphic operations over ciphertext after re-keying, the construction scheme is inefficient due to the use of a fully homomorphic encryption scheme and bootstrapping techniques. Recently Derler et al. [5] introduced a homomorphic proxy-authenticator, which addresses the efficiency gap in [8] by using ElGamal. However, their construction is bidirectional and cannot provide a one-way re-keying solution.

Polyakov et al. [6] introduced the first multi-hop unidirectional PRE scheme. Their scheme is efficient compared to the other existing PREs based on the fully homomorphic encryption (FHE) schemes. However, in general, FHEs are far more computationally expensive than partial homomorphic encryption (PHE) schemes. Bellafqira et al. [7] introduced another homomorphic proxy re-encryption based on the Damgard-Jurik cryptosystem [23]. In [7], authors introduced a re-encryption process based on computing the difference between encrypted data and encrypting the result with a new key. In

fact, the process of re-encryption in [7] does not directly embed the new key into the ciphertext tuple, but it decrypts and re-encrypts the ciphertext with a new key.

1.3. PRELIMINARIES

In this section, we explain the definitions stated in this paper, the encryption scheme used as the basis for our proposal, and provide the used symbols in Table V.1.1.

1.3.1. ONE-DIRECTION PROXY RE-ENCRYPTION

Let us assume a proxy re-encryption scheme with the following algorithms: (KeyGen, ReKeyGen, Encryption, ReEncryption, Decryption).

- KeyGen generates a pair of public and private keys $A = (pk, sk)$.
- ReKeyGen takes A and $\acute{pk}$ and generates a token $r_{A \Rightarrow B}$.
- Encryption algorithm takes pk and a message m and outputs $[m]_{pk}$.
- Decryption takes $[m]_{pk}$ or $[m]_{\acute{pk}}$ and outputs m.
- ReEncryption gets a ciphertext under A and changes its key using $r_{A \Rightarrow B}$ to compute $[m]_{\acute{pk}}$ without using Decryption. If $r_{A \Rightarrow B}$ cannot be used to convert $[m]_{\acute{pk}}$ to $[m]_{pk}$ the proxy encryption is one-directional.

1.3.2. CORRECTNESS OF PROXY RE-ENCRYPTION

To check the correctness of a HPRES, the validity of four properties should be analyzed:

1. Decryption$([m]_{pk}, sk) = m$,
2. Decryption((ReEnrypttion$([m]_{pk}, \acute{pk}), \acute{sk}) = m$,
3. Decryption$([m_1]_{pk} \oplus [m_2]_{pk}, sk) = m_1 + m_2$, and
4. Decryption$([m_1]_{\acute{pk}} \oplus [m_2]_{\acute{pk}}, \acute{sk}) = m_1 + m_2$.

1.3.3. PUBLIC-KEY CRYPTOSYSTEM WITH A DOUBLE TRAPDOOR DECRYPTION

Bresson et al. in [24] developed a public key cryptosystem BCP03 that is the basis for our scheme. The process of key generation, encryption, decryption in BCP03 are as follows:

- Key generation: Choose two large prime numbers $\acute{p}$ and $\acute{q}$, compute $p = 2\acute{p}+1$ and $q = 2\acute{q}+1$, and compute their product to obtain $N = pq$. Then, choose two random numbers α and a, where $\alpha \in \mathbb{Z}^*_{N^2}$ and $a \in [1, N\acute{p}\acute{q}]$ accordingly. Afterwards, g and h are computed as $g = \alpha^2 \bmod N^2$ and $h = g^a \bmod N^2$. The public key is (N, g, h) and the private key is a.
- Encryption: It takes a message m and the public key, and outputs $C = (A, B)$, where $A = g^r \bmod N^2$ and $B = h^r(1 + mN) \bmod N^2$.
- Decryption: It takes C and the secret key a, and outputs $m = (B/(A^a) - 1 \bmod N^2)/N$.

V.1

1.4. HOMOMORPHIC ONE-DIRECTION PROXY RE-ENCRYPTION SCHEME (HOPE)

Protocol 14 HOPE

1: KeyGen() $\Rightarrow (pk_X, lsk_X, sk_X)$
2: ReKeyGen$(pk_Y, lsk_Y, sk_X) \Rightarrow (rk^{(1)}_{X\to Y}, rk^{(2)}_{X\to Y}))$, where $rk^{(1)}_{X\to Y} = (\dot{A}, \dot{B})$, and $rk^{(2)}_{X\to Y} = a_X - \dot{\beta} \bmod (p_X q_X \acute{p}_X \acute{q}_X)$
3: Enc$(pk_X, m) \Rightarrow K_{pk_X} = (A, B)$
4: ReEnc$(rk^{(1)}_{X\to Y}, rk^{(2)}_{X\to Y}), K_{pk_X}) \Rightarrow \acute{K}_{pk_Y} = (A, \acute{A}, \dot{A}, B, \dot{B})$
5: Dec$(K_{pk_X}) \Rightarrow m = \frac{B/(A^a)-1 \bmod N_X^2}{N_X}$
6: Dec$(\acute{K}_{pk_Y}) \Rightarrow m = \frac{B/(\acute{A}\cdot A^{\dot{\beta}})-1 \bmod N_Y^2}{N_Y}$, where $\dot{\beta} = \frac{\dot{B}/(\dot{A})^b-1 \bmod N^2}{N}$.

1. KeyGen: Choose four primes p, q, $\acute{p}$, and $\acute{q}$ such that $p = 2\acute{p}+1$ and $q = 2\acute{q}+1$, then a safe-prime modulus $N = pq$. Let $\mathbb{G}$ to be a cyclic group of quadratic residues number of N^2. Then, choose three random numbers $a, b \in \{1, ord(\mathbb{G})\}$ and $\alpha \in N^2$, where $ord(\mathbb{G}) = \lambda(N^2)/2 = pq\acute{p}\acute{q} = N\lambda(N)/2$ with $\lambda(N) = 2\acute{p}\acute{q}$, and k_1 as a security parameter. Afterwards, set $g_0 = \alpha^2 \bmod N^2$, $g_1 = g_0^a \bmod N^2$, and $g_2 = g_0^b \bmod N^2$. The public key $pk = (N, g_0, g_1, g_2)$, the local secret key is $lsk = (a, b)$, and the master secret key is $sk = (p, q, \acute{p}, \acute{q})$.

2. ReKeyGen: Having the public key pk_Y, lsk_X, and the master secret key sk_X, it outputs the unidirectional re-encryption key $rk_{X\to Y} = (rk^{(1)}_{X\to Y}, rk^{(2)}_{X\to Y})$, where $rk^{(1)}_{X\to Y} = (\dot{A}, \dot{B})$. For the computation of $\dot{A}$ and $\dot{B}$, first a random number $\dot{\beta} \in \{0,1\}^{k_1}$. Then, compute $rk^{(2)}_{X\to Y} = a_X - \dot{\beta} \bmod (p_X q_X \acute{p}_X \acute{q}_X)$. Afterwards, compute $r_{X\to Y} = H_Y(\dot{\beta})$, $\dot{A} = (g_{Y0})^{r_{X\to Y}}$, and $\dot{B} = (g_{y2})^{r_{X\to Y}} \cdot (1 + \dot{\beta} N_Y) \bmod (N_Y)^2$.

3. Enc: It takes a public key pk and a message $m \in \{0,1\}^n$ and performs following operations: (1) choose a random number $r \in \mathbb{Z}_{N^2}$, (2) compute $A = (g_0)^r$, and (3) compute $B = (g_1)^r \cdot (1 + mN) \bmod N^2$. The final result is $K = (A, B)$

4. ReEnc: It takes a re-encryption key $rk_{X\to Y}$ and a ciphertext K under public key pk_X, then re-encrypt K to another ciphertext under public key pk_Y. ReEnc computes $\acute{A} = A^{rk^{(2)}_{X\to Y}} \bmod N_X^2$ and forms the tuple $\acute{K} = (\acute{A}, B, \dot{A}, \dot{B}, A)$.

5. Dec: There are two decryption functions: (1) for normal ciphertexts K, and (2) re-encrypted versions $\acute{K}$. To decrypt a K we simply compute

$$m = \frac{B/(A^a) - 1 \bmod N^2}{N}. \tag{1.1}$$

To decrypt a $\acute{K}$, we first compute

$$\dot{\beta} = \frac{\dot{B}/(\dot{A})^b - 1 \bmod N^2}{N}, \tag{1.2}$$

and then

$$m = \frac{B/(\acute{A} \cdot A^{\dot{\beta}}) - 1 \bmod N^2}{N}. \tag{1.3}$$

Table V.1.1: Table of symbols.

Symbol	Definition
a, b, α	random numbers
g_0	$\alpha^2 \bmod N^2$
g_1	$g_0^a \bmod N^2$
g_2	$g_0^b \bmod N^2$
pk	(N, g_0, g_1, g_2)
lsk	(a, b)
sk	$(p, q, \acute{p}, \acute{q})$
A	$(g_0)^r$
B	$(g_1)^r \cdot (1 + mN) \bmod N^2$
$\acute{A}$	$(g_0)^{r(a-\dot{\beta})} \bmod N^2$
$\dot{A}$	$(g_0)^{r_{X\to Y}}$
$\dot{B}$	$(g_2)^{r_{X\to Y}} \cdot (1 + \dot{\beta}N) \bmod N^2$

1.4.1. CORRECTNESS

DECRYPTION OF K

$$\begin{aligned} &\frac{B/(A^a) - 1 \bmod N^2}{N} \\ &= \frac{(g_1)^r \cdot (1 + mN)/(g_0)^{ar} - 1 \bmod N^2}{N} \\ &= \frac{1 + mN - 1 \bmod N^2}{N} = m \end{aligned} \tag{1.4}$$

DECRYPTION OF $\acute{K}$

To decrypt $\acute{K}$, first the value $\dot{\beta}$ is computes, then $\dot{\beta}$ is used for decryption of a re-encrypted message.

$$\begin{aligned} &\frac{\dot{B}/(\dot{A})^b - 1 \bmod N^2}{N} \\ &= \frac{(g_2)^{r_{X\to Y}} \cdot (1 + \dot{\beta}N)/(g_0)^{br_{X\to Y}} - 1 \bmod N^2}{N} \\ &= \frac{1 + \dot{\beta}N - 1}{N} = \dot{\beta} \end{aligned} \tag{1.5}$$

Next, we need to show that by using $\dot{\beta}$ we can decrypt a re-encrypted message m by computing

$$\begin{aligned}
&\frac{B/(\acute{A}\cdot A^{\dot{\beta}})-1 \bmod N^2}{N} \\
&= \frac{(g_1)^r\cdot(1+mN)/(g_0)^{r(a-\dot{\beta})+r\dot{\beta}}-1 \bmod N^2}{N} \\
&= \frac{(g_0)^{ar}\cdot(1+mN)/(g_0)^{ar}-1 \bmod N^2}{N} \\
&= \frac{1+mN-1}{N} = m
\end{aligned} \tag{1.6}$$

1.4.2. HOMOMORPHISM

$K_1 \cdot K_2$

We show that multiplication of two encrypted integer values $[a]\cdot[b]$ under HOPE yields the encryption of the addition of that two values, $[a+b]$. Let assume that $K_1 = (A_1, B_1)$ and $K_2 = (A_2, B_2)$, then $K_1 \cdot K_2 = (A_{12}, B_{12})$, where $A_{12} = (g_0)^{r_1+r_2}$ and

$$\begin{aligned}
B_{12} =& (g_1)^{r_1+r_2}\cdot(1+m_1N)\cdot(1+m_2N) \bmod N^2 \\
\equiv& (g_1)^{r_1+r_2}\cdot(1+(m_1+m_2)N) \bmod N^2.
\end{aligned} \tag{1.7}$$

Then, the decryption of $K_1 \cdot K_2$ equals to

$$\begin{aligned}
&\frac{B_{12}/(A_{12}^a)-1 \bmod N^2}{N} \\
&= \frac{(g_1)^{r_1+r_2}\cdot(1+(m_1+m_2)N)/(g_0)^{a(r_1+r_2)}-1 \bmod N^2}{N} \\
&= \frac{1+(m_1+m_2)-1}{N} = m_1+m_2.
\end{aligned} \tag{1.8}$$

$\acute{K}_1 \cdot \acute{K}_2$

HOPE enables homomorphism over re-encrypted version of ciphertexts. Let's $\acute{K}_1 \cdot \acute{K}_2 = (\acute{A}_{12}, B_{12}, \dot{A}_{12}, \dot{B}_{12})$, where $\dot{A}_{12} = \dot{A}_1 || \dot{A}_2$, $A_{12} = A_1 || A_2$, $B_{12} = (g_1)^{r_1+r_2}\cdot(1+(m_1+m_2)N)$, $\acute{A}_{12} = (g_0)^{r_1(a-\dot{\beta}_1)+r_2(a-\dot{\beta}_2)}$, and $\dot{B}_{12} = \dot{B}_1 || \dot{B}_2$. Similar to decryption of $\acute{K}$, first, we need to obtain $\dot{\beta}_1$ and $\dot{\beta}_2$, then we use them to decrypt the ciphertext.

$$\begin{aligned}
&\frac{\dot{B}_1/(\dot{A}_1)^b-1 \bmod N^2}{N} \\
&= \frac{(g_2)^{r_{X\to Y}}\cdot(1+\dot{\beta}_1N)/(g_0)^{br_{X\to Y}}-1 \bmod N^2}{N} \\
&= \frac{1+\dot{\beta}_1N-1}{N} = \dot{\beta}_1
\end{aligned} \tag{1.9}$$

By performing similar computation as in Equation 1.9, $\dot{\beta}_2$ is obtained based on the $\dot{A}_2$ and $\dot{B}_2$ values. Then, by using the Equation 1.3, we can obtain $m_1 + m_2$ as follow:

$$\begin{aligned} &\frac{B_{12}/(\acute{A}_{12} \cdot A_1^{\dot{\beta}_1} \cdot A_2^{\dot{\beta}_2}) - 1 \bmod N^2}{N} \\ &= \frac{\frac{(g_1)^{r_1+r_2} \cdot (1+(m_1+m_2)N)}{(g_0)^{r_1(a-\dot{\beta}_1)+r_2(a-\dot{\beta}_2)+r_1\dot{\beta}_1+r_2\dot{\beta}_2}} - 1}{N} \\ &= \frac{1+(m_1+m_2)N-1}{N} = m_1 + m_2 \end{aligned} \tag{1.10}$$

Note that if both re-encrypted ciphertexts $\acute{K}_1$ and $\acute{K}_2$ are originally encrypted under the same public key then $\dot{B}_1 = \dot{B}_2$.

K^v

In HOPE, the decryption of $K^v = (A, B)^v$, where v is an integer number, results in vm. In computation of $(A, B)^v$, $A^v = (g_0)^{vr}$ and B^v is computed as follows:

$$\begin{aligned} B^v &= ((g_1)^r \cdot (1 + mN))^v \bmod N^2 \\ &= (g_1)^{vr} \cdot (1 + vmN) \bmod N^2 \end{aligned} \tag{1.11}$$

The decryption of K^v is as follows:

$$\begin{aligned} &\frac{B/(A^a) - 1 \bmod N^2}{N} \\ &= \frac{(g_1)^{vr} \cdot (1 + vmN)/(g_0)^{var} - 1 \bmod N^2}{N} \\ &= \frac{1 + vmN - 1 \bmod N^2}{N} = vm \end{aligned} \tag{1.12}$$

1.4.3. DATA PACKING IN HOPE

Although, as it is shown in Section 1.6, HOPE protocol is more efficient than the state-of-the-art solution, applying data packing improves the performance of HOPE significantly concerning computation and communication. The main idea behind data packing is to efficiently use the message space of a Homomorphic encryption system in a protocol. There are two types of data packing (DP): 1) Concatenation-based DP (C-DP) [25], and 2) Subset Sum Problem-based DP (SSP-DP) [26]. In C-DP, assume K is the encryption of a ℓ-bit integer, and n is the message space of HOPE. Party A can pack $\rho = \lfloor \log n / \log \ell \rfloor$ different K_i into one HOPE encryption as follows:

$$[\hat{K}] = \sum_{i=0}^{\rho-1} [m]_i^{(2^\ell)^i}. \tag{1.13}$$

Afterwards, re-encryption and decryption can be performed on $\hat{K}$, instead of each K_i individually, which reduces computational cost. Another packing technique is based on SSP. SSP is a well-known NP-complete problem [27], where given a set $A = \{a_i : 1 \le i \le n\}$

of integers and another positive integer M, the problem is finding a subset of A has sum equal to M [28]. In SSP-DP, K_i are summed such that it is possible to extract every single K_i from the summed value later without knowing the original K_i. C-DP is more simple and straightforward to implement than SSP-DP; instead, SSP-DP uses the message space more efficiently and packs more K_i in single encryption.

1.5. SECURITY

In this section, we show that HOPE is semantically secure. First, we prove that ciphertext generated by HOPE cannot be decrypted without using the valid secret key. Second, we show that the re-encrypted version of ciphertext is still secure.

1.5.1. COMPUTATIONAL DIFFIE-HELLMAN PROBLEM (CDH)

The CDH assumption in $\mathbb{G}_1$ holds if for a generator of $\mathbb{G}_1$ and given g, g^a, and g^b, it is computationally hard to compute g^{ab}, where $a, b \in \mathbb{Z}_q$ and q is a prime number. In fact, the advantage of any polynomial time algorithms in solving CDH should be negligible. We show that HOPE is secure based on the CDH problem, where providing pk, A, B, $\acute{A}$, $\dot{A}$, $\dot{B}$, and $\dot{E}$ it is computationally difficult to compute g_0^{ar}.

1.5.2. LIFT DIFFIE-HELLMAN PROBLEM (LDH)

Let $g, A, B, C \in \mathbb{G}$ where $A = g^a \bmod N^2$, $B = g^b \bmod N^2$, and $C = g^{ab} \bmod N^2$. LDH is defined as the difficulty of computing Z, when A, B, N, g, and $Z \bmod N$ are given. It is proved in [] that the LDH problem is computationally difficult to solve as the computational class problem in Paillier [14].

Theorem 11. *It is computationally hard to compute g_0^{ar} given $K = (A, B)$ and pk for every probabilistic polynomial time algorithm $\mathscr{A}$ if there exists a negligible function f such that for sufficient large ℓ:*

$$\begin{aligned} Pr\Big(\mathscr{A}\big(N, K \bmod N^2, K \bmod N\big) = g_0^{ar} \bmod N^2 \,\big| \\ N = pq; \quad g_0, g_1, g_2; \quad A = (g_0)^r; B = (g_1)^r \cdot (1 + mN)\Big) \\ = f(\ell) \end{aligned} \tag{1.14}$$

Proof. As it is stated in Equation 1.14, we should consider that adversary has access to $B \equiv g_1^r \bmod N$, since N is publicly known. Given a ciphertext $K = (A, B)$ and public information (N, g_0, g_1, g_2), we need to prove that the adversary cannot learn about the message m without having the secret key. In order to retrieve m out of K, adversary should somehow obtain g_0^{ar} and put it in Equation 1.1. There are two possible ways for adversary to obtain g_0^{ar}: 1) by using $g_1 = g_0^a \bmod N^2$ and $A = g_0^r \bmod N^2$ to obtain $g_0^{ar} \bmod N^2$ and 2) reducing $B \bmod N^2$ to N, then computing $g_1^r \bmod N^2$ from $g_1^r \bmod N$. However, both ways are computationally hard based on the CDH and LDH problems. □

Theorem 12. *Given $\acute{K} = (\acute{A}, \dot{A}, B, \dot{B}, \dot{E})$ and pk, it should be computationally hard to compute g_0^{br} for every probabilistic polynomial time algorithm $\mathscr{A}$ if there exists a negligible function f such that for sufficient large ℓ*

$$\begin{aligned} Pr\Big(&\mathscr{A}(N, \acute{K} \bmod N^2, \acute{K} \bmod N) = g_0^{br} \bmod N^2 \big| N = pq \\ &g_0, g_1, g_2; \quad \acute{A} = (g_0)^{r(a_X - \dot{\beta})}; \quad B = (g_1)^r \cdot (1 + mN); \\ &\dot{A} = (g_0)^{r_{X\to Y}}; \quad \dot{B} = (g_2)^{r_{X\to Y}} \cdot (1 + \dot{\beta}N); \\ &\dot{E} = (g_0)^{\dot{\beta}(r-1)}\Big) = f(\ell) \end{aligned} \tag{1.15}$$

Proof. Similar to the proof for Theorem 11, adversary can reduce $\acute{K}$ to N, which gives $g_1^r \bmod N$ and $(g_2)^{r_{x\to Y}} \bmod N$. As it is shown in Equation 1.5, having $(g_2)^{r_{x\to Y}} \bmod N^2$ is enough to obtain $\dot{\beta}$ that can be used later to decrypt the $\acute{K}$. However, it is computationally hard to get $(g_2)^{r_{x\to Y}} \bmod N^2$ out of $(g_2)^{r_{x\to Y}} \bmod N$, because of the LDH problem.

Another way to decrypt $\acute{K}$ is to obtain $g_0^{\dot{\beta}}$ by computing $g_0^{\dot{\beta}r}$. Since the adversary has access to $g_1^r \bmod N$, if he could compute $g_1^r \bmod N^2$, then computation of $g_1^r / \acute{A}$ gives $g_0^{r\dot{\beta}} \bmod N^2$. Finally, computing $g_0^{r\dot{\beta}} / \dot{E}$ gives $g_0^{\dot{\beta}}$ that can be used to decrypt a ciphertext $\acute{K}$. However, it is proved that obtaining $g_1^r \bmod N^2$ from $g_1^r \bmod N$ is a LDH problem. □

Theorem 13. *No polynomial time distinguisher adversary $\acute{A}$ can break the semantic security of HOPE.*

Proof. Let's assume that there are two messages m_0 and m_1, and $b \in 0, 1$ is a random bit. We need to show that if $\acute{A}$ is given $B = (g_1)^r \cdot (1 + m_b N)$, he has no advantage on finding b than guessing its value randomly. Since g is a generator of the group $\mathbb{G}$, we can represent g^r, $r \in \mathbb{Z}_{N^2}$, as $g^{r_1 + r_2\lambda(N)}$, where $r_1, r_2 \in \mathbb{Z}_N$. Note that $g^{\lambda(N)/2} = (1 + \sigma N)$, where $\sigma \in \mathbb{Z}_N$. Then, we have

$$\begin{aligned} B &= (g_1)^r (1 + m_b N) \bmod N^2 = (g_1)^{r_1 + r_2\lambda(N)} (1 + m_b N) \\ &= g_1^{r_1} g_1^{r_2\lambda(N)} (1 + m_b N) = g_1^{r_1} (1 + \sigma N)^{r_2} (1 + m_b N) \\ &= g_1^{r_1} (1 + r_2 \sigma N)(1 + m_b N) \\ &= g_1^{r_1} (1 + (r_2\sigma + m_b)N) \bmod N^2 \end{aligned} \tag{1.16}$$

In Equation 1.16, it is shown that m_b is perfectly masked with a random number $r_2 \in \mathbb{Z}_N$. Thus, $\mathscr{A}$ cannot computationally distinguish between the encryption of m_0 and m_1. □

1.6. PERFORMANCE ANALYSIS

In this section, we present the performance analysis of our proposal. First, we compare HOPE with [7] in terms of computation and communication. Then, we deploy HOPE to enable secure data searching in a multiple-key setting using a secure search protocol (IBSv1) [10]. We use C++ and external libraries: MPIR, Boost, (a library to be made

explicit later) on a single Linux machine running Ubuntu 14.04 LTS, with a 64-bit microprocessor and 8 GB of RAM for the implementation. In order to compare our work with [7], we use HOPE for sharing an encrypted image of size $92 * 122$ pixels. In our implementation, we set the key size 1024 bits and the security parameter 120 bits as in [7]. We also apply C-DP, because of its simplicity, to our implementation to achieve better performance. Table V.1.2 shows the run-times of both protocols, where HOPE is 99.2% more efficient than [BCBQC17] in re-encrypting ciphertexts. Moreover, in HOPE, there is no computation to be performed by the delegatee during the re-encryption phase.

Table V.1.2: Comparing the run-times (second) of HOPE and [7], denoted as [BCBQC17], for sharing an image of $92 * 122$ pixels.

Protocols	Delegator	Proxy	Delegate
[BCBQC17]	0.004	120	30
HOPE	0.0001	0.96	0.0

Deploying data packing in HOPE also reduces the communication cost for sharing the image from 1.3 mega-bytes in [BCBQC17] to 0.1 mega-bytes in HOPE that is 92% improvement. Table V.1.3 shows more experimental information about sharing the image using HOPE.

Table V.1.3: Run-times (second) of encryption, packing, re-encryption, and decryption of sharing an image of $92 * 122$ pixels using HOPE.

Protocols	encryption	packing	re-encryption	decryption
HOPE	0.3	0.55	0.41	0.45

Next experiment is to use HOPE for secure data searching by using IBSvI [10]. Similar to [10], we also implement the secure data searching protocol up to step 4, as it is shown in Figure V.1.1, for the sake of comparison. We fix our parameters in the implementation according to Table V.1.4 as in [10].

Table V.1.4: Parameters and their values.

Parameter	Symbol	Value
Bit-size of inputs	ℓ	15 bits
Number of records in RCS	ω	10^5
Security parameter	κ	112 bits
Capacity of a package (IBSvI)	$\acute{\rho}$	89
Key size and message space	N	2048 bits

Table V.1.5 shows the computation and communication costs of IBSvI using Paillier encryption scheme [14] and HOPE. For better performance analysis of HOPE, we run our experiments in two settings of secure data searching: 1) single-key setting denoted as HOPE_1 and 2) multiple-key setting, HOPE_2. In HOPE_1, similar to [10], we assume that KM has the secret key for the decryption purpose; in other words, RCS does not need to

re-encrypt the encryptions before decryption. This assumption is to compare the performance of Paillier and HOPE in the same setting. However, in $HOPE_2$, we assume that RCS has the token/s to re-encrypt ciphertexts according to the KM's public key and re-encrypt ciphertexts before sending them to KM for decryption. Hence, in $HOPE_2$, KM does not have access to the data owners' secret key.

Table V.1.5: Run-times and communication costs of the IBSvI and HOPE.

Protocols	Run-time (second)		Data transmission (megabyte)	index size (megabyte)
	RCS	KM		
$IBSvI_{Paillier}$	0.52	5.06	2.19	8.2
$IBSvI_{HOPE_1}$	0.46	27.42	4.38	16.2
$IBSvI_{HOPE_2}$	14.06	28.43	7.665	16.2

According to the Table V.1.5, the computation cost of $HOPE_1$ in RCS is comparable with Paillier. However, the decryption cost of HOPE is more expensive that Paillier. In $HOPE_2$, RCS demands more computational resources, because each ciphertext is re-encrypted to the KM's public key. Since the computational cost of decryption of a re-encrypted ciphertext is more than a ciphertext in its original form, the run-times of IBSvI in KM are different in $HOPE_1$ and $HOPE_2$. In terms of communication cost, HOPE encryption consists of two ciphertexts, which makes it more expensive than Paillier. Note that after re-encryption the size of the encryption becomes larger which results in higher data transmission cost in $HOPE_2$.

1.7. CONCLUSION

In this work, we introduced a homomorphic proxy re-encryption scheme, HOPE. We showed that HOPE provides homomorphism after re-encryption of data. HOPE is also compatible with data packing, which is used to reduce the computational and communication overhead in cryptographic protocols. Our experimental results showed that HOPE is computationally 99% more efficient than the state-of-the-art HPRES. We also tested HOPE in a realistic scenario, where there is a database of 10^5 encrypted records under different publicly keys. Then, we showed that the overall computation for filtering the encrypted records with HOPE finishes in less than a minute, which proves the efficiency of its re-encryption function. These experimental results show that HOPE is a promising solution for HPRES, and a secure one, even with data sets used in real life.

REFERENCES

[1] A. Jeckmans, A. Peter, and P. Hartel, *Efficient privacy-enhanced familiarity-based recommender system*, in *Computer Security–ESORICS 2013* (Springer, 2013) pp. 400–417.

[2] A. Sadeghi, T. Schneider, and I. Wehrenberg, *Efficient privacy-preserving face recognition*, in *Information, Security and Cryptology - ICISC 2009, 12th International*

Conference, Seoul, Korea, December 2-4, 2009, Revised Selected Papers (2009) pp. 229–244.

[3] G. Couteau, *New protocols for secure equality test and comparison,* in *Applied Cryptography and Network Security - 16th International Conference, ACNS 2018, Leuven, Belgium, July 2-4, 2018, Proceedings* (2018) pp. 303–320.

[4] 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.

[5] D. Derler, S. Ramacher, and D. Slamanig, *Homomorphic proxy re-authenticators and applications to verifiable multi-user data aggregation,* in *Financial Cryptography and Data Security - 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers* (2017) pp. 124–142.

[6] Y. Polyakov, K. Rohloff, G. Sahu, and V. Vaikuntanathan, *Fast proxy re-encryption for publish/subscribe systems,* ACM Trans. Priv. Secur. **20**, 14:1 (2017).

[7] R. Bellafqira, G. Coatrieux, D. Bouslimi, G. Quellec, and M. Cozic, *Proxy re-encryption based on homomorphic encryption,* in *Proceedings of the 33rd Annual Computer Security Applications Conference, Orlando, FL, USA, December 4-8, 2017* (2017) pp. 154–161.

[8] C. Ma, J. Li, and W. Ouyang, *A homomorphic proxy re-encryption from lattices,* in *Provable Security - 10th International Conference, ProvSec 2016, Nanjing, China, November 10-11, 2016, Proceedings* (2016) pp. 353–372.

[9] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, *Improved proxy re-encryption schemes with applications to secure distributed storage,* ACM Trans. Inf. Syst. Secur. **9**, 1 (2006).

[10] M. Nateghizad, Z. Erkin, and R. L. Lagendijk, *Efficient index-based search protocols for encrypted databases,* in *Proceedings of the 15th International Joint Conference on e-Business and Telecommunications, ICETE 2018 - Volume 2: SECRYPT, Porto, Portugal, July 26-28, 2018.* (2018) pp. 436–447.

[11] M. Blaze, G. Bleumer, and M. Strauss, *Divertible protocols and atomic proxy cryptography,* in *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding* (1998) pp. 127–144.

[12] A. Ivan and Y. Dodis, *Proxy cryptography revisited,* in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2003, San Diego, California, USA* (2003).

[13] E. Ayday, J. L. Raisaro, J. Hubaux, and J. Rougemont, *Protecting and evaluating genomic privacy in medical tests and personalized medicine,* in *Proceedings of the 12th*

annual ACM Workshop on Privacy in the Electronic Society, WPES 2013, Berlin, Germany, November 4, 2013 (2013) pp. 95–106.

[14] 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.

[15] R. Canetti and S. Hohenberger, *Chosen-ciphertext secure proxy re-encryption,* in *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007* (2007) pp. 185–194.

[16] B. Libert and D. Vergnaud, *Unidirectional chosen-ciphertext secure proxy re-encryption,* IEEE Trans. Information Theory **57**, 1786 (2011).

[17] D. Nuñez, I. Agudo, and J. Lopez, *A parametric family of attack models for proxy re-encryption,* in *IEEE 28th Computer Security Foundations Symposium, CSF 2015, Verona, Italy, 13-17 July, 2015* (2015) pp. 290–301.

[18] D. Nuñez, I. Agudo, and J. Lopez, *On the application of generic cca-secure transformations to proxy re-encryption,* Security and Communication Networks **9**, 1769 (2016).

[19] K. Xagawa, *Cryptography with Lattices,* Master's thesis, Department of Mathematical and Computing Sciences Tokyo Institute of Technology (2010), available at http://xagawa.net/pdf/2010Thesis.pdf.

[20] E. Kirshanova, *Proxy re-encryption from lattices,* in *Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings* (2014) pp. 77–94.

[21] N. Chandran, M. Chase, F. Liu, R. Nishimaki, and K. Xagawa, *Re-encryption, functional re-encryption, and multi-hop re-encryption: A framework for achieving obfuscation-based security and instantiations from lattices,* in *Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings* (2014) pp. 95–112.

[22] C. Gentry, A. Sahai, and B. Waters, *Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based,* IACR Cryptology ePrint Archive **2013**, 340 (2013).

[23] I. Damgård and M. Jurik, *A length-flexible threshold cryptosystem with applications,* in *Information Security and Privacy, 8th Australasian Conference, ACISP 2003, Wollongong, Australia, July 9-11, 2003, Proceedings* (2003) pp. 350–364.

[24] E. Bresson, D. Catalano, and D. Pointcheval, *A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications,* in *Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings* (2003) pp. 37–54.

[25] J. R. Troncoso-Pastoriza, S. Katzenbeisser, M. U. Celik, and A. N. Lemma, *A secure multidimensional point inclusion protocol,* in *Proceedings of the 9th workshop on Multimedia & Security, MM&Sec 2007, Dallas, Texas, USA, September 20-21, 2007* (2007) pp. 109–120.

[26] M. Nateghizad, Z. Erkin, and R. L. Lagendijk, *A novel approach for data packing: Using trapdoor knapsack,* in *Proceedings of Conference: IEEE International Workshop on Information Forensics and Security, WIFS 2018 - Hong Kong, December 11-13, 2018.* (2018).

[27] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (W. H. Freeman, 1979).

[28] R. C. Merkle and M. E. Hellman, *Hiding information and signatures in trapdoor knapsacks,* IEEE Trans. Information Theory **24**, 525 (1978).

VI

Outlook

VI.1 | Discussion and Future Work

In this chapter, we reflect on the research questions and challenges in the privacy-preserving e-healthcare system as the objective of this thesis. We show how the challenges are addressed and what the future works are to progress toward more efficient and secure e-healthcare systems.

VI.1

1.1. DISCUSSION

In this section, we address the research questions in realizing real-life privacy-preserving e-healthcare scenarios, which are:

- How can changing system and security configuration in e-healthcare affect security and performance?
- How should the challenge of resource-demanding cryptographic applications be addressed effectively?
- Is it possible to achieve practical privacy-preserving e-healthcare systems using homomorphic encryption?

To address the first research question, we described three scenarios in chapter I.3, which have different system and security settings. Then, we investigated the challenges of developing each scenario to meet privacy and performance requirements. In order effectively address the challenge of resource demanding cryptographic applications, we analyzed the performance of several applications that are using core building blocks in Chapter I.4. We showed that the core building blocks are dominating the computation and communication costs of the applications.

Afterward, we improved the existing core building blocks, namely equality testing and comparison protocols, to minimize that performance gap in e-healthcare systems. We also developed new cryptographic building blocks and techniques, index-based searching and data packing protocols to reduce the number of homomorphic operations in the applications. Moreover, we introduced an efficient homomorphic proxy re-encryption scheme to address the challenge of processing data that are encrypted under different keys.

1.1.1. CORE BUILDING BLOCKS

One of the main tasks of the three scenarios in Chapter I.3 is performing secure data filtering in encrypted databases. Filtering can include Boolean operations, 'AND' and 'OR', comparison, and equality testing protocols. Thus, one of the objectives of this thesis is to improve the performance of existing cryptographic core building blocks, which includes secure equality testing and comparison protocols. In this section, we show our achievements in this regard.

EQUALITY TESTING PROTOCOL

Secure equality testing plays an important role in many cryptographic applications. This building block is mostly used in secure searching and data matching protocols. Therefore, we have developed a variety of equality testing based on the amount of available computational and communicational resources.

As the first try in achieving high-performance equality testing protocols, we improved two state-of-the-art protocols [1, 2] (LT13 and ST06), denoted as NEL-I and NEL-II, respectively. The objectives were to improve the computational cost of LT13 and reduce the decryption and data transmission costs of ST06. As it is shown in Chapter II.1, we improved the computational cost of LT13 from 331169 multiplications for 20-bit inputs

to 11861 multiplications in NEL-I that is 96% improvements. However, our computational improvements introduced 50% more data transmission cost and one more communication round. We also extensively used data packing to improve the performance of ST06, where the number of decryptions and the cost of data transmission are reduced by 50% and 54%, respectively, in NEL-II. In terms of the total run-time, we achieved 99% and 95% improvement for LT13 and ST06, respectively.

In our second work, we developed three building blocks in Chapter II.2, namely EQT-1, EQT-2, and EQT-3. The goal was to introduce highly efficient equality testing protocols with different balances between computation and communication costs. Among the three proposals, EQT-1 has the least data transmission cost and highest computation cost. EQT-1 has 48% less data transmission than LT13 and also less 12% computation cost than NEL-I. Therefore, EQT-1 is more efficient than NEL-I and LT13 in terms of computation and communication costs. Then, in EQT-2, we reduced the computational complexity from 10435 in EQT-1 to 4240 homomorphic multiplications, which is 59% improvement. However, that improvement resulted in 27% more data transmission cost than EQT-1. To improve the computation cost further, we introduced EQT-3. The number of multiplications in EQT-3 is reduced to 74 from 4240 in EQT-2 that is 98% improvement at the cost of more 38% data transmission cost than EQT-2. In total, we improved the state-of-the-art equality testing protocols computationally and communicationally by 99.98% and 24%, respectively.

To achieve a more communication-wise efficient equality testing protocol, we introduced another equality testing protocol, SET, in chapter II.3. We used different techniques such as garbled circuits, homomorphic encryption, data packing, etc. in our work. As a result, SET is communication-wise more efficient than EQT-1 by 70%. SET is communication and computation-wise more efficient than EQT-1 and EQT-2; however, its computation cost is 27% more than EQT-3.

In this thesis, we proposed techniques to decrease the number of homomorphic operations and the data transmission cost in testing the equality of two encrypted integers. Using our equality testing protocols, performing 10^6 equality tests takes about an hour in a typical desktop system and demands 3 Gigabytes data transmission. Although we achieved highly efficient equality testing protocols, the responsiveness of our equality testing protocols in large databases is far from real-time unless high performance computing servers are used. Moreover, we ignored the network delay in our experimental analysis, which can also be a challenge in real-life practice.

COMPARISON PROTOCOL

As we showed in Chapter I.4, there are many applications for secure comparison protocols. We also showed that existing comparison protocols have a significant contribution to overall computation and communication costs. In chapter III.1, we proposed improvements to decrease the costs of the comparison protocol by introducing algorithmic changes and using data packing. First, we introduced a new algorithm for comparing private inputs, which is introduced by Damgård et al. in [3, 4]. According to our experimental results, we achieved 91% improvement in reducing the computational cost. Moreover, by using data packing in our new comparison protocol, we reduced the computation overhead of operations that performed in the key manager by 85%. Our new

comparison protocol, equipped by data packing, resulted in reducing the computation costs of the state-of-the-art protocol for comparing two encrypted inputs by 56%.

We applied our comparison protocol on one of the works described in [5] to show its effect on total computation cost. We showed that in Chapter III.2 changing the comparison protocol used in [6] with our new protocol reduced the total computation cost significantly. This reduction of computational complexity reflected on the run-time and reduced it approximately by 80%.

Our experimental results show the high effectiveness of our improvements in decreasing the total cost of the applications that are using secure comparison protocol. However, there are still expensive operations that have been addressed yet. According to our experimental analysis, DGK zero-check [3, 4] takes 65% of the total computation cost of our comparison protocol. Although DGK zero-check is more efficient than performing decryption, it takes a significant amount of computational resources. In general, similar to the performance of our efficient equality testing protocols, the performance of our comparison protocol in terms of computation and communication is not yet suitable for large databases.

1.1.2. INDEX BASED DATA FILTERING

Although using highly efficient core building blocks significantly affect the performance of cryptographic applications, using them in large encrypted data sets results in enormous computation and communication costs. Assuming a database with 10^5 encrypted records, using EQT-3, as the most computation-wise efficient building block, for filtering data takes roughly 330 seconds in a normal desktop system. This is the run-time for the execution of a query that has a single condition for filtering. The run-time of a query with two conditions with a 'AND' or 'OR' operation is doubled, 660 seconds. As an example, a query could be filtering the patients' records who have Blood Pressure (BP) or Heart Rate (HR) higher than a Threshold (T). This query can be represented as:

$$((device_{type} == BP)\ AND\ (BP > T_1))\ OR\ ((device_{type} == HR)\ AND\ (HR > T_2)) \quad (1.1)$$

According to the query in Equation 1.1, there are two equality testings, two comparisons, two 'AND' operations, and one 'OR' operation.

To address this issue, we developed a novel index-based data filtering and searching, IBSvI in Chapter IV.1. IBSvI makes an encrypted index of the encrypted data. Having the index, we can execute queries for data filtering with multiple conditions and 'OR' operation with only one communication round. Moreover, IBSvI does not use any building block, but only secure zero check. Using IBSvI, execution of a query with a single condition on a database with 10^5 encrypted records takes only 5.5 seconds that is 98% more efficient than using EQT-3. Moreover, the communication cost of executing that query is reduced from 276 MB in SET to 2.19 MB in IBSvI, which is 99% improvement. Recall that SET is most communication-wise efficient equality testing protocol. In term of data transmission cost and index size, IBSvI is 91% and 99.99% more efficient than the state-of-the-art [7], respectively, at the cost of 37.5% more computation cost.

To decrease the computation cost of IBSvI, we developed another index based filtering IBSvII by using a fully homomorphic encryption scheme and extensive use of data

packing. As a result, we reduced the communication round from one to zero, which makes the data transmission cost zero as well. IBSvII is computation-wise 84% and 77% more efficient than IBSvI and [7], respectively. However, the index size of IBSvII is 90% more than IBSvI, but it is 99.8% less than [7]. In terms of preserving the confidentiality of data, unlike the work in [7] both IBSvI and IBSvII hide access pattern, while data are being filtered.

Although our protocols for secure indexing and query execution are more efficient than previous works, there are a few limitations with our proposals. Our protocols do not support 'AND' operation and execution of a query that has both 'AND' and 'OR' operation is a challenge. Moreover, packing multiple encrypted measurements to achieve efficient indexing and query execution protocols makes the process of changing an encrypted measurement in a package computationally expensive.

1.1.3. DATA PACKING

As it is shown in Chapter I.4, packing encrypted data for storing, processing, or decrypting ciphertexts can significantly reduce the computation and communication costs of cryptographic applications. In Chapter IV.2, we introduced a novel data packing technique based on the Subset Sum Problem (SSPDP) for the first time. Unlike the existing data packing, our solution packs the data based on addition, and not concatenation (CDP). The main benefit of SSP based data packing is that packing encrypted zero does not take any space in the package, which results in more efficient data packing regarding the capacity of the package. Using SSP based data packing in data filtering significantly increases the performance in terms of computation and communication. The result of data filtering is a vector mostly filled with encrypted zeros and a few non-zero encrypted integers; thus, being able to efficiently pack non-zero encrypted integers save lots of data transmission and storage. To achieve efficiency in different settings, we used several techniques to achieve SSPDP, namely super increasing subset sum problem, multiplicative trapdoor knapsack, and compact knapsack problem. We also found that combining CDP and SSPDP increases the package capacity when the inputs are small numbers.

As it is shown in Chapter IV.2, the computation cost of packing 10^6 encrypted integers using SSPDP is decreased by 96% in data filtering. Moreover, the data transmission and decryption costs of filtering and retrieving the result using SSPDP is 99.98% and 99.99% less than using existing data packing, respectively. This significant improvement is the result of being able to pack only encrypted non-zero integers in a set of records filled by encrypted zero and non-zero integers.

Although our SSPDP-based data packing techniques introduce a new approach in reducing the cost of cryptographic applications, they can be deployed efficiently only in limited cases. For packing a set of encrypted data in a package using SSPDP-based data packing, the set should be mostly filled by encrypted zeros. Otherwise, its performance becomes similar to the performance of concatenation-based data packing.

1.1.4. HOMOMORPHIC PROXY RE-ENCRYPTION SCHEME

One of the main challenges in the scenario-2 is processing data that are encrypted under different public keys. Proxy re-encryption schemes are introduced to avoid decrypting the ciphertexts one-by-one to be encrypted under the new key. Therefore, Proxy

re-encryption schemes save a significant amount of computation and communication costs. In chapter V.1, we introduced a novel homomorphic proxy re-encryption scheme, HOPE. To the best of our knowledge, HOPE is the only scheme that has all of the following properties:

- Ciphertexts have homomorphic property before and after re-encryption.
- It is a one-direction re-encryption scheme.
- Re-encryption token does not leak any private information.
- Re-encrypted ciphertexts can be re-encrypted an unlimited number of times, and they have homomorphic property.
- Re-encrypted token is generated in a non-interactive form.

We used HOPE to filter and retrieve data in a setting where they are encrypted under different keys. To do so, we also used IBSvI for data filtering and SSPDP for data packing. Our experimental results show that filtering data using HOPE in a multiple key setting increases the computation, data transmission, and index size costs by 87%, 71%, and 49%, respectively. We also compared HOPE with the most similar state-of-the-art homomorphic proxy re-encryption [8]. Our experimental results show that HOPE outperforms [8] in terms of token generation and re-encryption costs by 97.5% and 99%, respectively.

Although HOPE is more efficient than the other homomorphic proxy re-encryption schemes, its computation cost is still high. According to our analysis in chapter V.1, if we use HOPE as a homomorphic encryption scheme in scenario 1 (single-key setting), instead of Paillier encryption scheme [9], the performance degrades roughly by five times.

1.2. FUTURE WORK

In this section, we described the challenges in the three scenarios, which have not addressed in this thesis, as future works.

1.2.1. SCENARIO-1

- Efficient additive homomorphic encryption scheme: Our analysis shows that the costs of encryption and decryption in our optimized core building blocks dominate the total computation cost. As an example, the run-time of performing one equality testing using EQT-3 is roughly 3.3 milliseconds. However, performing 74 homomorphic multiplications, as the computation complexity of EQT-3 excluding encryption and decryption costs, takes around 0.6 milliseconds. This difference shows that 82% of the total run-time of EQT-3 is dominated by encryption and decryption. Thus, developing an efficient additive homomorphic encryption scheme can have a significant effect on decreasing the computation costs of the core building blocks.
- Efficient zero-check: As we have shown in Chapter I.4, DGK zero-check dominates the computation cost of our comparison protocol. Thus, developing an efficient zero-check mechanism can effectively reduce the computation cost of our comparison protocol.

- Data filtering: In our index based searching protocol, our protocol supports 'AND' operation in the query; however, 'OR' and 'NOT' operations have remained unsupported. The indexing and filtering phases should be developed such that they can execute different operations in a query.

- Medical recommendations: Although we improved a privacy-preserving recommender system using our building block, more recommender systems should be developed to check the efficiency of our core building blocks in practice.

1.2.2. SCENARIO-2

- Data filtering: Removing the key manager from scenario-2 necessities shifting the computational demanding process more to the cloud side. That is because DSPs do not have much computation resources. Therefore, we need to develop core building blocks and other protocols to meet that requirement. Moreover, we need to develop an indexing and filtering technique that can support a database of encrypted data under different keys.

- Data retrieving: Since in this scenario DSPs are responsible to perform joint operations with the cloud, it is necessary to investigate a new approach for searching in this setting. On the one side, DSPs can help to develop more efficient indexing techniques, since they have access to their data in clear. On another side, DSPs should not learn whom the medical institute is looking for.

- Medical recommendations: The challenge of generating statistics and recommendations in a multiple-key setting is not addressed in this thesis. Thus, investigation through possible secure and efficient solutions is required.

1.3. CONCLUSION

Developing highly efficient e-healthcare systems play an important role in improving public health, while it can save a tremendous amount of healthcare costs. However, the privacy concerns regarding the leakage of privacy-sensitive data are the main drawbacks against putting them in large-scale practice. Moreover, to comply with GDPR, developing technological solutions to protect confidential data is necessary. Among several technological solutions, data encryption is one of the widely used technique in the literature to fill the privacy gap in e-healthcare systems.

In this thesis, we identified and addressed technical challenges in realizing secure and efficient e-healthcare systems. To do so and target the effective challenges carefully, we introduced three real-life e-healthcare scenarios. Then, based on the scenarios, we identified cryptographic tools that are required to realize the scenarios. Afterward, by analyzing several cryptographic applications, we showed that which cryptographic protocols are dominating the computation and communication costs. After that, we proposed improvements and developed new protocols to minimize computation and communication costs. Thus, our answer to the last research question of this thesis, '**Is it possible to achieve practical privacy-preserving e-healthcare systems using homomorphic encryption?**', is 'yes!'. Even though the costs of the applications based on our building blocks are still high (if it is compared with unsecured versions of the applications), using

powerful computing servers and applying optimizations such as parallel computing to the implementations help to achieve real-time privacy-preserving e-healthcare systems.

Although our achievements significantly improve the performance of e-healthcare systems, the communication and computation costs of secured e-healthcare systems using data encryption is far more than the costs in unsecured versions. Therefore, further investigation to improve and develop more efficient cryptographic building blocks and protocols is necessary. Moreover, it is necessary to evaluate the performance of e-healthcare systems in the covert and malicious security settings. We expect a significant increase in computation and communication costs of cryptographic protocols in the covert and malicious security settings. Therefore, developing efficient e-healthcare systems, which are secure against covert and malicious adversaries, are still open challenges.

REFERENCES

[1] 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.

[2] B. Schoenmakers and P. Tuyls, *Efficient binary conversion for paillier encrypted values,* in *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings* (2006) pp. 522–537.

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

[4] I. Damgård, M. Geisler, and M. Krøigaard, *A correction to 'efficient and secure comparison for on-line auctions',* IJACT **1**, 323 (2009).

[5] D. Verhaert, M. Nateghizad, and Z. Erkin, *An efficient privacy-preserving recommender system for e-healthcare systems,* in *Proceedings of the 15th International Joint Conference on e-Business and Telecommunications, ICETE 2018 - Volume 2: SECRYPT, Porto, Portugal, July 26-28, 2018.* (2018) pp. 354–365.

[6] T. R. Hoens, M. Blanton, A. Steele, and N. V. Chawla, *Reliable medical recommendation systems with patient privacy,* ACM TIST **4**, 67:1 (2013).

[7] F. Krell, G. F. Ciocarlie, A. Gehani, and M. Raykova, *Low-leakage secure search for boolean expressions,* in *Topics in Cryptology - CT-RSA 2017 - The Cryptographers' Track at the RSA Conference 2017, San Francisco, CA, USA, February 14-17, 2017, Proceedings* (2017) pp. 397–413.

[8] R. Bellafqira, G. Coatrieux, D. Bouslimi, G. Quellec, and M. Cozic, *Proxy re-encryption based on homomorphic encryption,* in *Proceedings of the 33rd Annual Computer Security Applications Conference, Orlando, FL, USA, December 4-8, 2017* (2017) pp. 154–161.

[9] 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.