Reliable and Privacy-preserving Iris Remote Authentication Techniques

Submitted in partial fulfillment of the requirements

for the award of the degree of

DOCTOR OF PHILOSOPHY

Submitted by

Morampudi Mahesh Kumar

(Roll No. 701634)

Under the guidance of

Dr. Munaga V N K Prasad

and

Dr. U. S. N. Raju

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL

TELANGANA - 506004, INDIA

July 2020

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL
TELANGANA - 506004, INDIA

THESIS APPROVAL FOR Ph.D.

This is to certify that the thesis entitled, Reliable and Privacy-preserving Iris Remote Authentication Techniques, submitted by Mr. Morampudi Mahesh Kumar [Roll No. 701634] is approved for the degree of DOCTOR OF PHILOSOPHY at National Institute of Technology Warangal.

Examiner

Research Supervisor
Dr. Munaga V N K Prasad
Center for Affordable Technologies
Institute for Development &
Research in Banking Technology
India

Research Supervisor
Dr. U. S. N. Raju
Dept. of Computer Science and Engg.
NIT Warangal
India

Chairman
Prof. P. Radha Krishna
Head, Dept. of Computer Science and Engg.
NIT Warangal
India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL
TELANGANA - 506004, INDIA

CERTIFICATE

This is to certify that the thesis entitled, Reliable and Privacy-preserving Iris Remote Authentication Techniques, submitted in partial fulfillment of requirement for the award of degree of DOCTOR OF PHILOSOPHY to National Institute of Technology Warangal, is a bonafide research work done by Mr. Morampudi Mahesh Kumar [Roll No. 701634] under our supervision. The contents of the thesis have not been submitted elsewhere for the award of any degree.

Research Supervisor
Dr. Munaga V N K Prasad
Center for Affordable Technologies
Institute for Development &
Research in Banking Technology
India

Hyderabad
Date: 11-12-2020

Research Supervisor
Dr. U. S. N. Raju
Dept. of Computer Science and Engg.
NIT Warangal
India

Warangal
Date: 11-12-2020

DECLARATION

This is to certify that the work presented in the thesis entitled "*Reliable and Privacy-preserving Iris Remote Authentication Techniques*" is a bonafide work done by me under the supervision of Dr. Munaga V N K Prasad and Dr. U. S. N. Raju. The work was not submitted elsewhere for the award of any degree.

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/date/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Morampudi Mahesh Kumar
(Roll No. 701634)
Date: 11-12-2020

ACKNOWLEDGMENTS

"We could never learn to be brave and patient, if there were only joy in the world" - Helen Keller

"Difficulties in your life do not come to destroy you but to help you realise your hidden potential and power. Let difficulties know that you too are difficult" - A. P. J Abdul Kalam

"Setting goals is the first step in turning the invisible into the visible" - Tony Robbins

Every day during my Ph.D. has been a great opportunity for learning. This thesis work is the result whereby I have been supported by many people. It is a pleasant aspect that I have now the opportunity to express my gratitude for all of them.

First and foremost, I would like to thank the Lord almighty for glorifying me with all the strength and health to carry out my research work. It is with great pleasure that I acknowledge my sincere thanks and deep sense of gratitude to my supervisors Dr. Munaga V N K Prasad, Associate Professor, Institute for Development and Research in Banking Technology (IDRBT), Hyderabad and Dr. U. S. N. Raju, Assistant Professor, Department of Computer Science and Engineering, National Institute of Technology (NIT) - Warangal for their valuable guidance throughout the course. Their technical perception, profound knowledge, sincere effort in guiding a student and devotion to work have greatly charmed and constantly motivated me to work towards the goal. They always gave me ample time for discussions, reviewing my work and suggesting requisite corrections.

I extend my gratitude to the Doctoral Scrutiny Committee (DSC) members comprising of Prof. S. G. Sanjeevi, Dr. P. Venkata Subba Reddy, Dr. G. Siva Kumar for their insightful comments and suggestions during oral presentations. I am also thankful to Prof. B. B. Amberker for his presence with valuable suggestions during the presentation of this research work. I am lucky to attend lectures by Prof. B. B. Amberker during my tenure. I am immensely thankful to Dr. Ch. Sudhakar, Prof. R. B. V. Subramanyam and Prof. P. Radha Krishna Heads of Dept. of CSE and chairmans of DSC, during my tenure for providing adequate facilities in the department to carry out the oral presentations.

I wish to express my sincere thanks to Prof. N.V. Ramana Rao, Director, NIT Warangal and Dr. A. S. Ramasastri, Director, IDRBT, Hyderabad for providing the infrastructure and

facilities to carry out the research. I am also very much grateful to the faculty members of Computer Science and Engineering Department, and IDRBT for their moral support throughout my research work.

On the personal level, I would also like to thank my scholar friends in IDRBT and NIT-Warangal for their valuable suggestions and for extending selfless cooperation. Lastly, my gratitude to my family for their unconditional love, support and prayers for my success in achieving the goal.

Morampudi Mahesh Kumar

Dedicated to

My Parents

ABSTRACT

With the vast increase in the usage of biometric recognition, template protection for biometrics captured attention in the recent years. Since biometrics are irrevocable, it is very important to protect its privacy. Biometric template protection schemes such as cancelable biometrics, biometric cryptosystem and homomorphic encryption (HE) are introduced to provide privacy-preserving (PP) biometric authentication. PP biometric authentication enables a user to verify him or herself without sending the original biometric information to a server. HE is the most widely explored research area to construct PP biometric authentication system due to the advantages over cancelable biometrics and biometric cryptosystem. Most of the existing PP biometric authentication systems using HE assumed that the server performs computations honestly. In a malicious server setting, the server may return an arbitrary result to save the computational resources results in false accept/false reject.

This thesis focuses to solve the modify templates, intercept channel and override comparator attacks of biometric recognition system. A PP iris authentication system using Fan-Vercauteren scheme (PIAHC) is proposed to solve the modify templates and intercept channel attacks. In PIAHC, the rotational inconsistencies occurred due to the head tilt of a person are eliminated. A procedure to compute the hamming distance between the encrypted reference and probe templates is designed. Experimental results proves the efficiency of PIAHC. Blockchain-based multi-instance iris authentication (BMIAE), secure and verifiable multi-instance iris authentication using public auditor (SviaPA), secure and verifiable multi-instance iris authentication using Blockchain (SviaB), secure and verifiable machine-learning based iris authentication (SvaS) and multi-instance iris remote authentication using private multi-class perceptron on malicious cloud server (MIRAMCS) methods are proposed to provide privacy to the iris templates and also to check the correctness of the comparator result.

ElGamal and Paillier HE provides the confidentiality of the iris templates in BMIAE and SviaPA/SviaB. Fan-Vercauteren HE scheme provides the confidentiality of the iris templates in SvaS and MIRAMCS. The correctness of the comparator result is ensured by a public auditor in SviaPA, SvaS and MIRAMCS. The Blockchain provides the integrity of

the encrypted reference iris templates as well as trust of the comparator result in BMIAE, SviaB. The challenges of using Blockchain in biometrics are also addressed in BMIAE, SviaB. SvaS performs both training and classification of nearest neighbor and multi-class perceptron classification algorithms on encrypted data to provide privacy not only to the iris templates but also to the model. Multi-biometric systems use information from multiple sources to provide better recognition than unimodal biometric systems. So, the features of both left and right irises of a person are fused in BMIAE, SviaPA, SviaB. Finally, a feature-level fusion technique, contradistinguish similarity analysis (CSA) that minimizes the between-class correlations and maximizes the pair-wise correlations is proposed in MIRAMCS. Extensive experimental results on benchmark iris databases demonstrate that the proposed methods provides privacy to the iris templates with no loss in accuracy as well as trust of the comparator result.

Contents

ACKNOWLEDGMENTS i

ABSTRACT iv

List of Figures xii

List of Tables xvi

List of Algorithms xviii

List of Notations xix

Glossary xxi

1 Introduction 1

1.1 Biometric Recognition System (BRS) 2

1.2 Modes of Operation of a BRS 3

1.3 Attacks on Biometric Systems 3

1.3.1 Attacks on Template Databases 4

1.3.2 Attacks on Modules 4

1.3.3 Attacks on User Interfaces 4

1.3.4 Attacks on Channels between Modules 5

1.4 Biometric Template Protection and its Evolution 6

1.5 Desirable Properties of Biometric Template Protection techniques . . . 7

1.6 Taxonomy of Biometric Template Protection Schemes 7

1.6.1 Cancelable Biometrics 7

1.6.2 Biometric cryptosystems . . . 9
1.6.3 Hybrid Methods . . . 10
1.6.4 Homomorphic Encryption . . . 11
1.7 Properties, Functions and Categories of Homomorphic Encryption . . . 11
1.7.1 Properties of Homomorphic Encryption . . . 12
1.7.2 Functions of Homomorphic Encryption . . . 13
1.7.3 Categories of Homomorphic Encryption Schemes . . . 14
1.7.3.1 Partial Homomorphic Encryption (PHE) . . . 14
1.7.3.2 Somewhat Homomorphic Encryption (SHE) . . . 15
1.7.3.3 Fully Homomorphic Encryption (FHE) . . . 16
1.8 Information Fusion in Biometrics . . . 16
1.9 Performance of a Biometric System . . . 18
1.10 Benchmark Databases . . . 19
1.11 Motivation for present work, Aim & Objectives . . . 20
1.11.1 Aim . . . 20
1.11.2 Objectives . . . 20
1.12 Overview of the Contributions of the Thesis . . . 21
1.13 Thesis Organization . . . 22

2 Literature Survey **23**
2.1 Homomorphic Encryption applied to Biometric Authentication . . . 23
2.2 Machine Learning approaches applied to Iris Recognition . . . 26
2.3 Machine Learning on Encrypted Data . . . 29
2.4 Blockchain for Biometrics . . . 30
2.5 Resources and Analysis . . . 36
2.6 Summary . . . 38

3 Privacy-preserving Iris Authentication on Honest-but-Curious Server (PI-AHC) **39**
3.1 Generation of Iris Codes . . . 41
3.1.1 Generation of rotation-invariant iris template . . . 41

3.1.2 Compression of rotation-invariant iris template 43
3.1.3 Encode the compressed rotation-invariant iris template using Batching scheme 44
3.2 Ensuring the confidentiality of iris templates and Computation of Hamming Distance 45
3.2.1 Ensuring the confidentiality of iris templates 45
3.2.1.1 Key Generation 46
3.2.1.2 Encryption 46
3.2.1.3 Evaluation (Add & Multiply) 47
3.2.1.4 Decryption 48
3.2.2 Computation of Hamming Distance on encrypted data 49
3.3 Implementation details and Security Analysis of PIAHC 50
3.3.1 Performance Evaluation of PIAHC 51
3.3.2 Security Analysis of PIAHC 54
3.3.3 Computational Analysis of PIAHC 56
3.3.4 Comparison Analysis 59
3.4 Summary 59

4 Privacy-preserving Multi-Instance Iris Authentication on Untrusted Cloud Server using PHE schemes 60

4.1 BMIAE: Blockchain-based Multi-Instance Iris Authentication using Additive ElGamal Homomorphic Encryption 61
4.1.1 Generation of Integer vector from iris templates 62
4.1.1.1 Fusion of left and right iris template 63
4.1.1.2 Compression of fused iris template 63
4.1.1.3 Mapping of compressed iris template to integer vector 64
4.1.2 Ensuring the Confidentiality of Iris templates using ElGamal Homomorphic Encryption 66
4.1.2.1 Key Generation 66

4.1.2.2 Encryption . . . 67
4.1.2.3 Evaluation (Add) . . . 68
4.1.2.4 Decryption . . . 69
4.1.3 Ensuring the integrity of encrypted reference templates and trust on computed distance using Smart contract . . . 70
4.1.3.1 Ensuring the integrity of encrypted reference iris templates . . . 71
4.1.3.2 Encrypted distance computation in the Blockchain 71
4.1.4 Computation of Hamming distance (from $\varepsilon(s)$) (HDM) . . . 72
4.1.5 Limitations of BMIAE . . . 73
4.2 SviaPA: Secure and Verifiable Multi-Instance Iris Authentication using Public Auditor . . . 73
4.2.1 Preliminaries and Assumptions of SviaPA . . . 74
4.2.1.1 Autoencoder . . . 74
4.2.1.2 Assumptions . . . 76
4.2.2 Fusion & Reducing the dimensions of Iris code using Autoencoder 77
4.2.3 Ensuring Confidentiality for the Iris templates using Paillier Homomorphic Encryption . . . 77
4.2.3.1 Key Generation . . . 78
4.2.3.2 Encryption . . . 79
4.2.3.3 Evaluation . . . 80
4.2.3.4 Decryption . . . 80
4.2.4 Encrypted Distance Computation & Verifying the Correctness of Result . . . 81
4.2.4.1 Encrypted Distance Computation . . . 82
4.2.4.2 Verifying the Correctness of Result . . . 82
4.2.5 Limitations of SviaPA . . . 87
4.3 SviaB: Secure and Verifiable Multi-Instance Iris Authentication using Blockchain . . . 87
4.4 Implementation details and Security Analysis . . . 89

4.4.1 Performance Evaluation of BMIAE, SviaPA and SviaB 91
4.4.2 Security Analysis of BMIAE, SviaPA and SviaB 93
4.4.3 Computational Analysis of BMIAE, SviaPA and SviaB 100
4.4.3.1 Computational cost in terms of time & cost 100
4.4.3.2 Computational cost in terms of number of operations 101
4.4.4 Comparison Analysis of BMIAE, SviaPA and SviaB with existing methods 101
4.5 Summary 105

5 Privacy-preserving Machine Learning based Iris Authentication on untrusted Cloud Server using FHE Scheme 106
5.1 Preliminaries 107
5.1.1 Classification in machine learning algorithms: 107
5.1.1.1 Nearest Neighbor (NN): 107
5.1.1.2 Multi-class Perceptron (MCP): 108
5.2 SvaS: Secure and Verifiable Machine Learning based Iris Authentication System 109
5.2.1 Generation of Iris Code 109
5.2.2 Secure and Verifiable Machine Learning Classification 111
5.2.2.1 Private Nearest Neighbor 111
5.2.2.2 Verification Scheme for Nearest Neighbor 114
5.2.2.3 Private Multi-class Perceptron (PMCP) 117
5.2.2.4 Verification Scheme for Multi-class Perceptron . . 121
5.3 Multi-instance Iris Remote Authentication using private multi-class perceptron on Malicious Cloud Server (MIRAMCS) 124
5.3.1 Contradistinguish Similarity Analysis (CSA) 126
5.3.2 Fusion & Encoding 130
5.3.3 Verification Procedure 130
5.4 Implementation details and Security Analysis of SvaS, MIRAMCS . . . 133
5.4.1 Performance Evaluation of SvaS and MIRAMCS 133

5.4.2 Security Analysis of SvaS & MIRAMCS 138
5.4.3 Computational Analysis of SvaS & MIRAMCS 139
5.4.4 Comparison Analysis of SvaS & MIRAMCS 142
5.5 Summary 143

6 Conclusion and Future Scope **144**
6.1 Conclusions 144
6.2 Future Scope 146

Author's Publications **147**

Bibliography **149**

List of Figures

1.1 (a) Traditional authentication systems (based on what he knows and what he possess?) (b) Biometric based authentication systems based on who he is intrinsically 2

1.2 Block diagram of Biometric Recognition System 3

1.3 Possible attacks of Biometric Recognition System adapted from [1]. Rounded Rectangle and ellipse represent the modules and attacks of the Biometric system. 5

1.4 Evolution of Biometric Template Protection Schemes 6

1.5 A hierarchical taxonomy of Biometric Template Protection schemes 8

1.6 Framework of cancelable biometrics 9

1.7 Framework of Key Generation Biometric Cryptosystem 10

1.8 Framework of Key Binding Biometric Cryptosystem 10

1.9 Scenario explaining the advantage of homomorphic encryption 11

1.10 Additive Property of homomorphic encryption 12

1.11 Multiplicative Property of homomorphic encryption 13

1.12 Functions involved in homomorphic encryption 14

1.13 Evolution of Partial homomorphic encryption Schemes before Gentry's work [2] 14

1.14 Evolution of Somewhat homomorphic encryption Schemes before Gentry's work [2] 16

1.15 Multiple sources of evidence used for fusion 17

2.1 Percentage of Homomorphic Encryption schemes applied to each biometric trait 36

2.2 Percentage of each Homomorphic Encryption category applied to biometric recognition . . . 37
2.3 Percentage of machine learning classification techniques achieving PP training, PP classification and both discussed in the literature . . . 37

3.1 Block diagram of Privacy-preserving Iris Authentication on Honest-but-Curious Server (PIAHC) . . . 40
3.2 Compression of Bits($1 \times 10240 \rightarrow 1 \times 2560$) . . . 44
3.3 Key Generation function in BFV Homomorphic Encryption . . . 46
3.4 Encryption function in BFV Homomorphic Encryption . . . 47
3.5 Evaluation function in BFV Homomorphic Encryption . . . 48
3.6 Decryption function in BFV Homomorphic Encryption . . . 49
3.7 ROC Curves of PIAHC for CASIA-V 1.0, CASIA-V3-Interval, IITD and SDUMLA-HMT databases . . . 52
3.8 Genuine and Imposter distributions of PIAHC for (a) CASIA-V 1.0 (b) CASIA-V3-Interval (c) IITD and (d) SDUMLA-HMT databases . . . 53
3.9 EER, Separability Measures (d′ and KS test) for CASIA-V 1.0, CASIA-V3-Interval, IITD and SDUMLA-HMT databases . . . 54

4.1 BlocK diagram of Blockchain-based Multi-Instance Iris Authentication using Additive ElGamal Homomorphic Encryption (BMIAE). . . . 62
4.2 Compression of Bits in Blockchain-based Multi-Instance Iris Authentication using Additive ElGamal Homomorphic Encryption . . . 65
4.3 Comparison of Equal Error Rate for various sizes of iris template . . . 65
4.4 Key Generation function in ElGamal Homomorphic Encryption . . . 67
4.5 Encryption function in ElGamal Homomorphic Encryption . . . 68
4.6 Evaluation function in ElGamal Homomorphic Encryption . . . 68
4.7 Decryption function in ElGamal Homomorphic Encryption . . . 69
4.8 Contract-Ensuring the Integrity of Reference templates and Trust on Distance Computation (EIRTDC) . . . 70

4.9 Block diagram of SviaPA. The dashed line, Dotted line and Solid line indicates the steps during enrollment, after the enrollment and during the authentication phases. 74
4.10 Key Generation function in Paillier Homomorphic Encryption 79
4.11 Encryption function in Paillier Homomorphic Encryption 79
4.12 Evaluation function in Paillier Homomorphic Encryption 80
4.13 Decryption function in Paillier Homomorphic Encryption 81
4.14 Block diagram of SviaB. The dashed line and Solid line indicates the steps during enrollment and the authentication phases. 89
4.15 DET curves of BMIAE for (a) CASIA-V3-Interval, (b) IITD, (c) SDUMLA-HMT databases 94
4.16 EER, Separability Measures (d′ and KS test) of BMIAE for CASIA-V3-Interval, IITD and SDUMLA-HMT databases 95
4.17 DET curves of SviaPA or SviaB for (a) CASIA-V3-Interval, (b) IITD, (c) SDUMLA-HMT databases 95
4.18 EER, Separability Measures (d′ and KS test) of SviaPA or SviaB for CASIA-V3-Interval, IITD and SDUMLA-HMT databases 96
4.19 Genuine and Imposter distributions of BMIAE for (a) CASIA-V3-Interval (b) IITD and (c) SDUMLA-HMT databases 97
4.20 Genuine and Imposter distributions of SviaPA/SviaB for (a) CASIA-V3-Interval (b) IITD and (c) SDUMLA-HMT databases 98

5.1 Perceptron for Multi-class Classification. $w_{i:1}, w_{i:2}, ..., w_{i:d}$ is the weight vector for the i^{th} class and $x_1, x_2, ..., x_d$ is the feature vector. 108
5.2 Block diagram of Secure and Verifiable Machine Learning based Iris Authentication System (SvaS) 110
5.3 Comparison of accuracy between iris code of sizes 10240, 2560, 1280 and 640 for MCP and NN 113
5.4 Homomorphic computation of Manhattan distance between vectors when vectors are encoded using batching scheme 114

5.5 Homomorphic computation of dot product between vectors when vectors are encoded using batching scheme 119
5.6 Block diagram of Multi-instance Iris Remote Authentication using private multi-class perceptron on Malicious Cloud Server (MIRAMCS) 125
5.7 Enrollment Phase of Multi-instance Iris Remote Authentication using private multi-class perceptron on Malicious Cloud Server (MIRAMCS) 126
5.8 Authentication Phase of Multi-instance Iris Remote Authentication using private multi-class perceptron on Malicious Cloud Server (MIRAMCS) . . 127
5.9 Comparison of before-normalization (BN) and after-normalization (AN) accuracies with different train-test split ratios (S1, S2 and S3 are described in Section 5.4.1) for a) CASIA-V3-Interval b) IITD database 131
5.10 Accuracy of SvaS (PMCP & PNN with different train-test split ratio) obtained for a) CASIA-V 1.0 b) CASIA-V3-Interval c) IITD d) SDUMLA-HMT iris databases 134
5.11 Comparison of accuracy of SvaS between protected and unprotected templates for a) MCP b) NN; **DB1**: CASIA-V 1.0, **DB2**: CASIA-V3-Interval, **DB3**: IITD & **DB4**: SDUMLA-HMT 135
5.12 Average classification accuracy of MIRAMCS obtained for a) CASIA-V3-Interval b) IITD iris databases 136
5.13 Comparison of accuracy of MIRAMCS between protected and unprotected templates 137

List of Tables

1.1 Non-invertible transforms vs Salting 9

2.1 Summary of selected works under Homomorphic encryption schemes applied to biometric recognition. 25
2.2 Summary of selected works under machine learning techniques applied to iris recognition. 27
2.3 Summary of selected works under machine learning classification techniques applied on encrypted data. 31
2.4 Active Research Communities for Biometric Template Protection schemes . 33
2.5 Some open-source FHE implementations. 34
2.6 Publicly available databases on which the methods in the literature are evaluated and their source. 35

3.1 Comparison of EER between original iris template and rotation-invariant iris template for CASIA-V 1.0 43
3.2 Compression of 10240 vector into blocks of various sizes for CASIA-V 1.0 44
3.3 Comparison of EER (in terms of %) between unprotected rotation-variant, unprotected rotation-invariant and protected rotation-variant iris template . . 51
3.4 Total time taken in PIAHC (with & without batching scheme) 56
3.5 Comparison Analysis in terms of Time 57
3.6 Comparison of PIAHC with existing approaches (EER in terms of %) . . . 58
3.7 Comparison of PIAHC with other approaches (in terms of Separability measure (d')) 58

4.1 EER obtained for databases CASIA-V3-Inerval, IITD and SDUMLA-HMT with different sizes of iris template 78
4.2 EER obtained in unprotected system for BMIAE, SviaPA and SviaB 91
4.3 EER obtained in protected system for BMIAE, SviaPA and SviaB 91
4.4 Baseline Comparison in terms of Storage cost (in Kilo Bytes (KB)), EER and Time (Average in seconds) for BMIAE 92
4.5 Baseline Comparison in terms of Storage cost (in Kilo Bytes (KB)), EER and Time (Average in seconds) for SviaPA and SviaB 93
4.6 Computational cost (for encryption and decryption (Average in secs) 101
4.7 Computation cost and time taken by smart contract to perform each operation. (BMIAE and SviaB considered a gas price of 3 gwei (1 gwei = 10^{-9} ETH) and 1 ETH = $176.83 for BMIAE (Real world values at time of writing, September 03, 2019) and 1 ETH = $239.15 for SviaB (Real world values at time of writing, March 07, 2020) 102
4.8 Computational cost in terms of number of operations 103
4.9 Comparison of BMIAE, SviaPA & SviaB with existing approaches (EER in terms of %) 103
4.10 Comparison of BMIAE, SviaPA & SviaB with other approaches (in terms of Separability measure (d')) 104
4.11 Comparison of biometric template protection schemes with BMIAE, SviaPA & SviaB 104

5.1 Accuracy obtained in unprotected system for MIRAMCS (MCP classifer with 80-20 train-test ratio) 137
5.2 Total time taken in SvaS (with & without batching scheme) 140
5.3 Total time taken in MIRAMCS (for n = 4096 & x = 40961) 141
5.4 Comparison Analysis in terms of Accuracy for CASIA-V3-Interval & IITD database 142
5.5 Comparison of biometric template protection schemes with SvaS and MIRAMCS 142

List of Algorithms

3.1 Enrollment phase of PIAHC 41
3.2 Authentication phase of PIAHC 42
3.3 Computation of Hamming distance on encrypted data 50
4.1 Enrollment phase of BMIAE 63
4.2 Authentication phase of BMIAE 64
4.3 Enrollment Phase of SviaPA 75
4.4 Authentication Phase of SviaPA 76
4.5 Generation of Encrypted Verification Vector in SviaPA 84
4.6 Correctness of result in SviaPA 85
4.7 Enrollment Phase of SviaB 90
4.8 Authentication Phase of SviaB 90
5.1 Enrollment Phase of SvaS 111
5.2 Authentication Phase of SvaS 112
5.3 Nearest Neighbor on Encrypted data (PNN) 114
5.4 Nearest Neighbor_Verification Vector 115
5.5 Nearest Neighbor_Correctness 116
5.6 Homomorphic Comparison (cmpsn) 118
5.7 Perceptron for multi-class classification on encrypted data (PMCP:training) 120
5.8 Perceptron for Multi-class Classification on encrypted data (PMCP:classification) 121
5.9 Multi-class Perceptron_Verification Vector 122
5.10 Multi-class Perceptron_Correctness 123

List of Notations

X_i	Reference compressed iris template/ Reference fused compressed iris template
Y	Compressed probe iris template/ Fused compressed probe iris template
id	Identifier of the end-user
d	Dimension of iris/fused iris template
M	Dimension of compressed/fused compressed iris template
N	Number of reference iris templates
P_k, S_k	Public and secret keys
$Enc(P_k,\ X_i)$	The encryption of X_i with P_k
$\varepsilon(msg)$	Encrypted value of msg
x	Modulus in the plain-text space (Plaintext Modulus)
n	A power of 2
q	Modulus in the cipher-text space
$a^n + 1$	The polynomial modulus which specifies the ring R
R	The ring $\mathbb{Z}[a]/(a^n + 1)$
R_x	The ring $\mathbb{Z}_x[a]/(a^n + 1)$ i.e., same as the ring R but with coefficient reduced modulo x
ω	A base into which ciphertext elements are decomposed during relinearisation
χ	Error distribution (a truncated Gaussian distribution)
Hr, Hp	Hash values computed in Blockchain
Q, g	Prime number, generator used in ElGamal homomorphic encryption

$\varepsilon(s)$	Computed distance between the encrypted reference and encrypted probe templates by server/Blockchain
$\varepsilon(Z_{n+1})$	Encrypted verification vector generated by trusted authenticator
$\varepsilon(V)$	Encrypted random vector generated by trusted authenticator. $\varepsilon(V)$ = ($\varepsilon(v1)$, $\varepsilon(v2)$, . . , $\varepsilon(v_N)$), where $\varepsilon(v_i)$ is the encrypted random integer
c	Number of classes i.e., subjects
w_i	Weight vectors
$\varepsilon(R)$	Computed manhattan distances in the case of nearest neighbor and dot products in the case of multi-class perceptron
l	class label of the training instance
E_l, E_r	Reference left and right iris templates
R_l, R_r	Reference left and right transformed iris templates
T_l, T_r	Transformation matrices
Q_l, Q_r	probe left and right iris templates
P_l, P_r	probe left and right transformed iris templates
H	Classification function used in the authentication phase $H = max_{i \in [0,\ N]} \sum_{j=1}^{d} (w_{ij}.y_j)$
(RA_k, RB)	Random data chosen by client device. It should be refreshed periodically.
$\varepsilon(RND)$	The encrypted classification result on random data (RA_k, RB) with the classification function H.

Glossary

BMIAE	Blockchain-based Multi-instance Iris Authentication using Additive ElGamal Homomorphic Encryption
BRS	Biometric Recognition System
BTP	Biometric Template Protection
CPT	Compressed Protected Template
CSA	Contradistinguish Similarity Analysis
CUT	Compressed Unprotected Template
DCA	Discriminant Correlation Analysis
Dec	Decryption
DET	Detection Error Tradeoff
EER	Equal Error Rate
Enc	Encryption
Eval	Evaluation
FAR	False Accept Rate
FHE	Fully Homomorphic Encryption
FMR	False Match Rate
FNMR	False Non-Match Rate
FRR	False Reject Rate
GAR	Genuine Accept Rate
HE	Homomorphic Encryption
KeyGen	Key Generation
MBS	Multi-Biometric Systems
MCP	Multi-Class Perceptron

MIRAMCS	Multi-instance Iris Remote Authentication using private multi-class perceptron on Malicious Cloud Server
NN	Nearest Neighbor
PIAHC	Privacy-preserving Iris Authentication on Honest-but-Curious Server
PIN	Personal Identification Number
PHE	Partial Homomorphic Encryption
PMCP	Private Multi-Class Perceptron
PNN	Private Nearest Neighbor
PP	Privacy-preserving
PRI	Protected Rotation Invariant
ROC	Receiver Operating Characteristic
SHE	Somewhat Homomorphic Encryption
SviaB	Secure and Verifiable Multi-Instance Iris Authentication using Blockchain
SviaPA	Secure and Verifiable Multi-Instance Iris Authentication using Public Auditor
SvaS	Secure and Verifiable Machine Learning based Iris Authentication System
URI	Unprotected Rotation Invariant
URV	Unprotected Rotation Variant
UUT	Uncompressed Unprotected Template

Chapter 1

Introduction

Person recognition plays a vital role in many applications. Examples of such applications include distributing social welfare benefits, granting access to nuclear facilities, performing remote financial transactions, managing international border crossings. The essential task in person recognition is to create an association between personal identity and an individual. A person can be recognized in three ways [3], as shown in Figure 1.1: (i) Knowledge-based (ii) Token-based (iii) Biometric Recognition. Knowledge-based recognition recognizes the person based on What he knows (Password, Cryptographic key or Personal Identification Number (PIN)). Token-based recognition recognizes the person based on What he possesses extrinsically (Passport, Identification Card, Driving License, etc). Biometric recognition recognizes the person based on What he does (Behavioral modalities) or Who he is intrinsically (Physiological modalities) [4].

Formally, biometric recognition can be described as the science of establishing the identity of a person based on the physiological or/and behavioral attributes of the person either in a semi-automated or fully automated manner [5]. Biometric authentication system demands the person to be present during the time of authentication, thus prevents the need to remember a password or carry a token. As a result, the identity of the user is difficult to lose, forge, duplicate or forgotten [3, 5, 6]. Fingerprint, iris, face, palmprint, etc. are the most used physiological biometric modalities. Gait, signature, keystroke dynamics are commonly used behavioral biometric modalities [5, 7]. These modalities are unique for a person results in uniqueness, permanence, and non-repudiation of biometrics [6, 8].

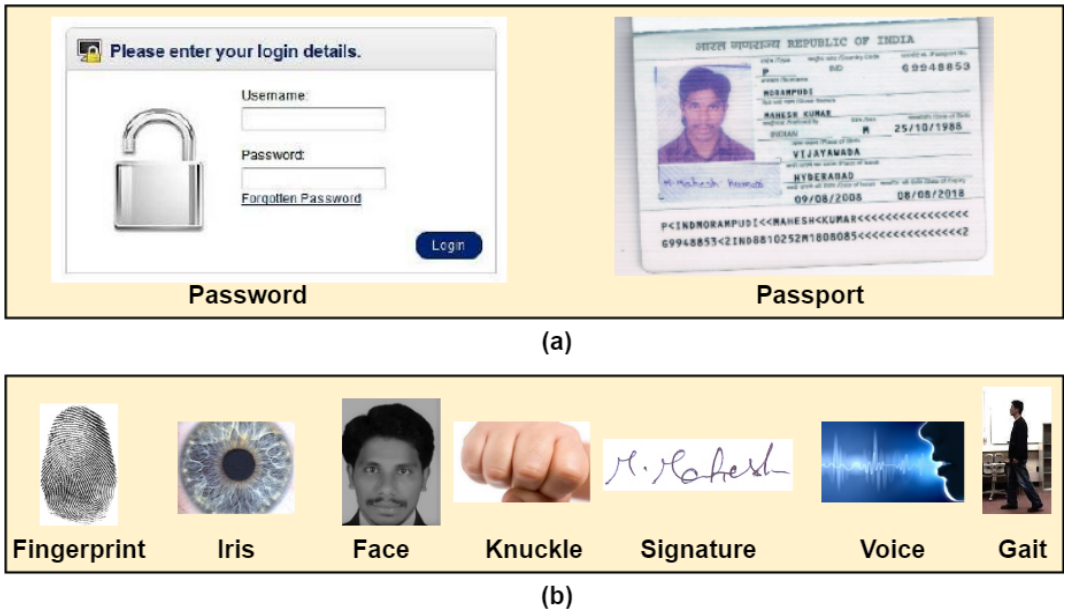


Figure 1.1: (a) Traditional authentication systems (based on what he knows and what he possess?) (b) Biometric based authentication systems based on who he is intrinsically

1.1 Biometric Recognition System (BRS)

BRS includes two phases, namely the enrollment phase & identification/verification phase [9] as shown in Figure 1.2. It also consists of five modules, namely sensor, feature extractor, template generator, comparator & decision module. The sensor helps to acquire the biometric characteristics from the person. During the acquisition of a biometric, there may be the possibility of unwanted background information, the occurrence of noise, etc. Therefore, preprocessing is needed to remove them. The unwanted background information can be removed with segmentation. The noise can be removed with filters. The second module, feature extractor plays a significant role in the BRS. Depending on the biometric trait and application, the number of features varies. The first time a person uses a biometric system is called enrollment. The feature extractor module extract features from reference biometric trait. The template generator module converts the extracted features into template and stored in the database. The same procedure is followed to extract features from the probe biometric at the time of the identification/verification phase. The comparator module compares the probe template with the reference template and produces the result to the decision module. The decision module provides a match (accept) or non-match (reject).

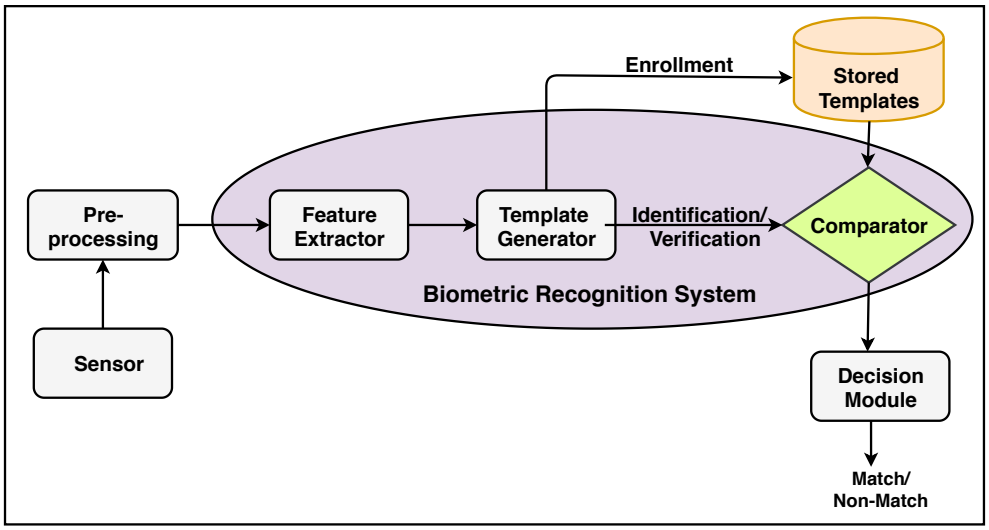


Figure 1.2: Block diagram of Biometric Recognition System

1.2 Modes of Operation of a BRS

BRS can be operated in two basic modes: verification & identification [7]. The system implements 1-to-1 comparison in the verification (authentication) mode. The probe template is compared with a specific template stored in the database to verify the individual is the person he/she claim to be. In verification mode, identity of a person like ID number (e.g. PIN), user name, or a smart card is used to indicate which template should be used for comparison. The system performs a one-to-many comparison to establish the identity of an unknown individual in the identification mode. The probe template is compared with all the reference templates stored in the database and produce a match (accept) result for an individual if the value falls within a predefined threshold; otherwise, it produces a non-match.

1.3 Attacks on Biometric Systems

The eight attack points of the biometric system found by Ratha *et al.* [1] are shown in Figure 1.3. Based on the type of attack, these attacks are categorized into four groups,

namely attacks on template databases, on modules, on user interfaces, channels between modules.

1.3.1 Attacks on Template Databases

The reference templates are stored in the database, either locally or remotely. Adversaries read these templates and modify/replace them results in the authorization for an intruder. The following vulnerabilities can be performed by an adversary with a stolen template:

- The adversary can replace an imposter's template with a template in the database results in false acceptance.
- The adversary can present the stolen template to the comparator module to gain unauthorized access.
- The adversary can create a physical spoof from the stolen template to access the system in an unauthorized way.

So, these type of attacks are considered to be the most dangerous attacks [6, 10].

1.3.2 Attacks on Modules

The attacks happening on the modules of BRS falls under this category. Spoofing, device substitution, coercive attacks are the possible attacks on the sensor module [10]. In a spoofing attack, an intruder enters into the system by using the genuine user biometric results in false accept. Device substitution attack refers to replacing the sensor device with the genuine capture device. The original biometric is presented to the sensor illegally, leads to coercive attack. The hacker produces the chosen feature sets by attacking the feature extractor module with a Trojan horse. The intruder attacks the comparator module and produces a fake score [11].

1.3.3 Attacks on User Interfaces

The fake biometric attack falls under this category. In this, the intruder presents a fake biometric such as the mask of a face, gummy finger to enter into the system at the sensor

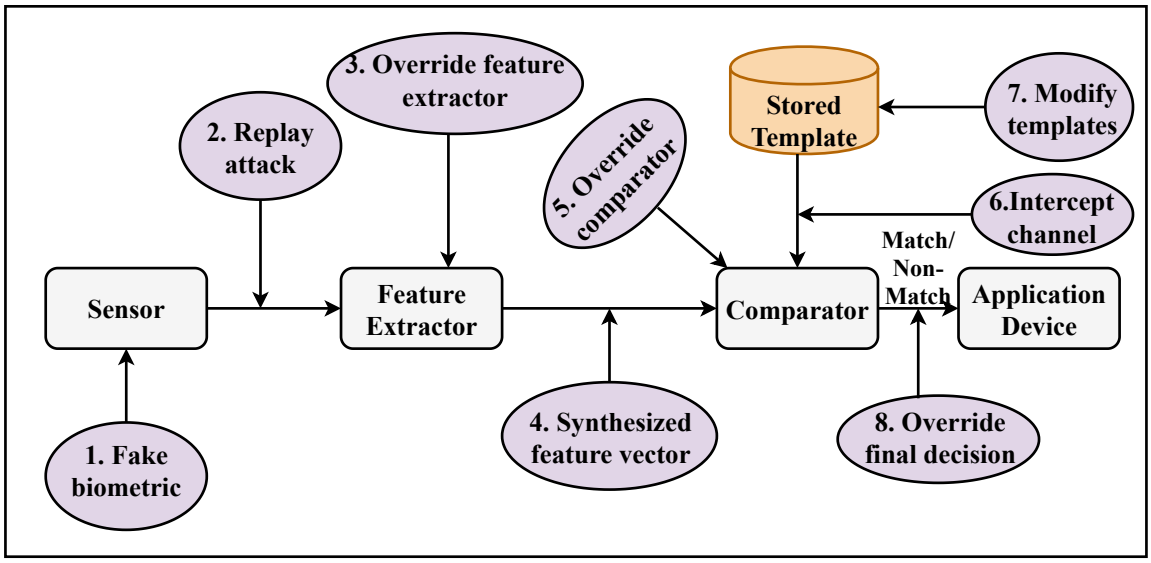


Figure 1.3: Possible attacks of Biometric Recognition System adapted from [1]. Rounded Rectangle and ellipse represent the modules and attacks of the Biometric system.

module [12]. The imposter enters into the system with a false identity when the sensor is not able to differentiate between the genuine and fake biometric modalities. New sensing technologies such as Touch-less, High resolution and Multi-spectral sensors should be used to overcome the fake biometric attack. Liveliness detection is also a possible solution to overcome this attack [13, 14, 15].

1.3.4 Attacks on Channels between Modules

The intruder may interrupt the channel between the modules results in a Replay attack, Synthesized feature vector, Intercept channel, and Override final decision [10, 11]. Replay attack refers to presenting the already recorded biometric data into the system instead of the acquisition of biometric data through the sensor. The originally extracted features are replaced with different synthesized features results in synthesized feature vector attack. This attack is tough to happen if the comparator and feature extractor are inseparable. On the other hand, in cloud-based authentication systems, the chance of this attack is real. Intercept channel attack refers to changing the contents of templates by the adversary during the transmission from the database to the comparator module. Override final decision attack refers to overriding the result produced by the comparator with the result of hacker's

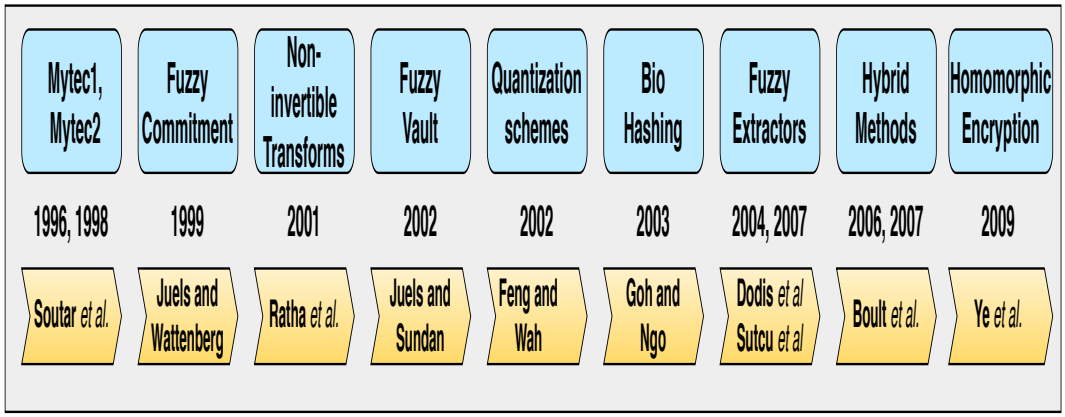


Figure 1.4: Evolution of Biometric Template Protection Schemes

choice, result in false accept [6, 16].

1.4 Biometric Template Protection and its Evolution

The vast increase in the usage of BRS in various applications has raised privacy and security concerns [6, 16]. As the biometric data is unique to a person, it is irrevocable if it gets compromised. Initially, it is believed that biometric data cannot be reconstructed from the extracted template. But studies in the literature such as [17, 18] proved that, an iris biometric could be reconstructed from iris template. In [19], the authors showed that a face biometric could be reconstructed from its template. The unauthorized access to biometric templates that are stored in the database results in several attacks like hill-climbing [20], replay, masquerade [21], and the stole-token attack [11], which makes the system vulnerable. Leakage or disclosure of biometric data to unauthorized persons causes the consequence of "Lose it once, it's gone forever". So, a biometric system capable of protecting the biometric templates need to be designed to ensure the privacy & security for user's data [22]. The evolution of Biometric Template Protection (BTP) schemes is shown in Figure 1.4.

1.5 Desirable Properties of Biometric Template Protection techniques

A BTP scheme should satisfy the following requirements according to ISO/IEC standard 24745 about Biometric Information Protection [23]:

1. **Diversity**: The protected templates used in various applications must not have any correlation. This ensures user's privacy (ISO/IEC 24745: 2011).

2. **Revocability**: The BTP scheme should be capable of canceling a compromised template and generate a new template (ISO/IEC 24745: 2011).

3. **Irreversibility**: The template generated by protection method must be non-invertible (Original template cannot be obtained from the secured template) (ISO/IEC 24745: 2011)

4. **Performance**: The accuracy of the recognition system should not be degraded due to the BTP scheme [11].

1.6 Taxonomy of Biometric Template Protection Schemes

BTP schemes are categorized into four types [22], namely cancelable biometrics, biometric cryptosystems, hybrid methods & homomorphic encryption as shown in Figure 1.5.

1.6.1 Cancelable Biometrics

A one-way transformation function is used to protect the biometric templates in cancelable biometrics [24]. The transformation function depends on a parameter, known as a key. In cancelable biometrics, matching or comparison is made between transformed reference & probe templates instead of original reference & probe templates, as shown in Figure 1.6. Cancelable biometrics can be categorized into Salting & Non-invertible transforms. Table 1.1 describes the differences between salting and non-invertible transforms.

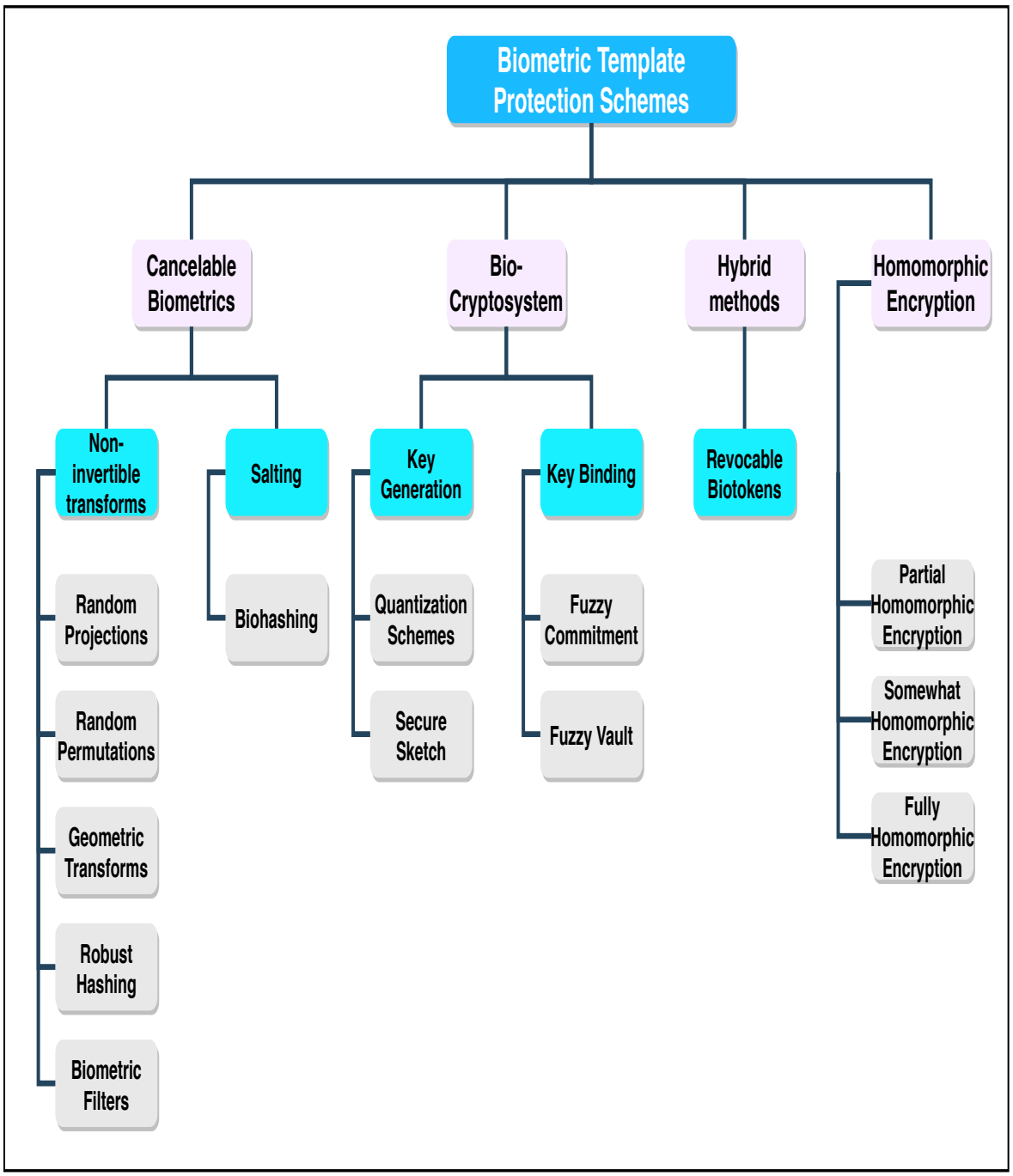


Figure 1.5: A hierarchical taxonomy of Biometric Template Protection schemes

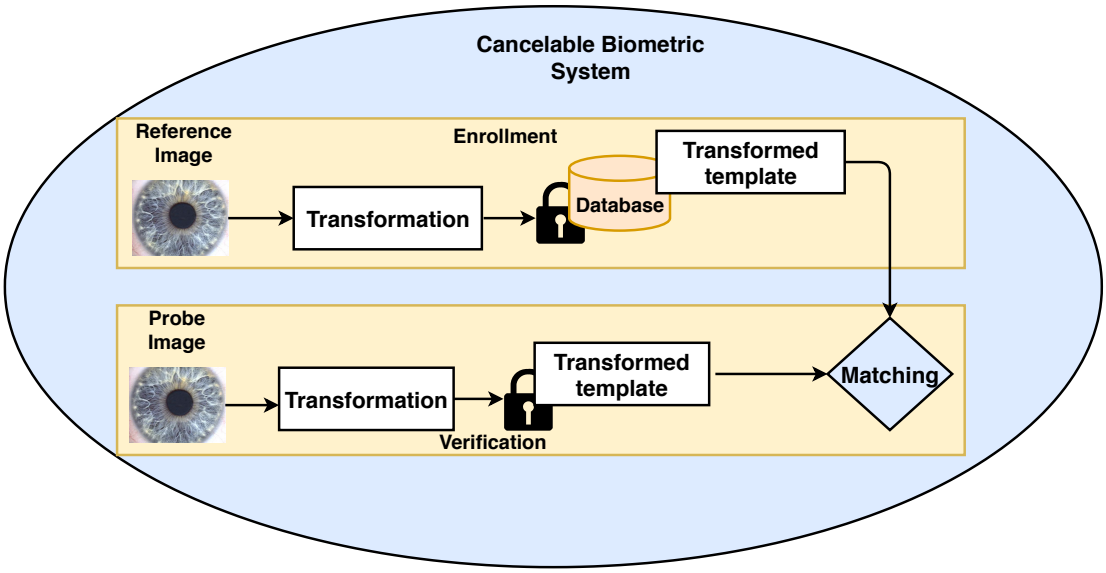


Figure 1.6: Framework of cancelable biometrics

Table 1.1: Non-invertible transforms vs Salting

Non-invertible transforms	**Salting**
The irreversible function is used as a transformation function.	An invertible function is used as a transformation function.
The key is produced during the authentication phase.	The key should be recalled or stored securely by the user during the authentication phase.
Unlike salting, the key is generated during the authentication phase as a result increases security, but with a loss of accuracy [10].	Once the key is lost, the intruder recovers original template results in permanent loss of biometric data.

1.6.2 Biometric cryptosystems

Biometric cryptosystems refer to generating a key or binding a key from or to a biometric feature [25]. The helper data is used to generate or bind keys. The biometric cryptosystems are categorized into Key Generation and Key Binding systems depending on how the helper data is obtained. The keys are directly generated from the biometric features in key generation cryptosystem, which is shown in Figure 1.7. One of the examples for key generation cryptosystem is Quantization scheme. The key is bound with the biometric feature to generate the helper data in a key binding cryptosystem, which is shown in Figure 1.8. Fuzzy Commitment [26] & Fuzzy Vault [27] schemes come under this category.

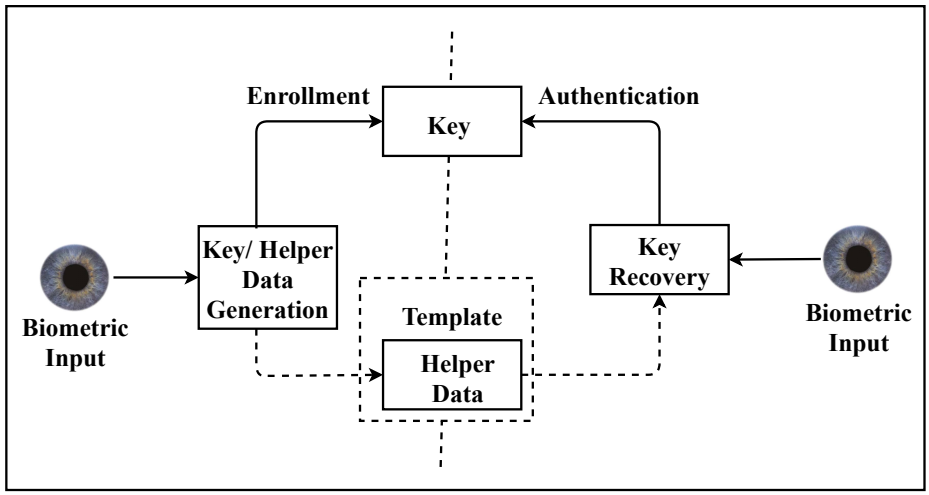


Figure 1.7: Framework of Key Generation Biometric Cryptosystem

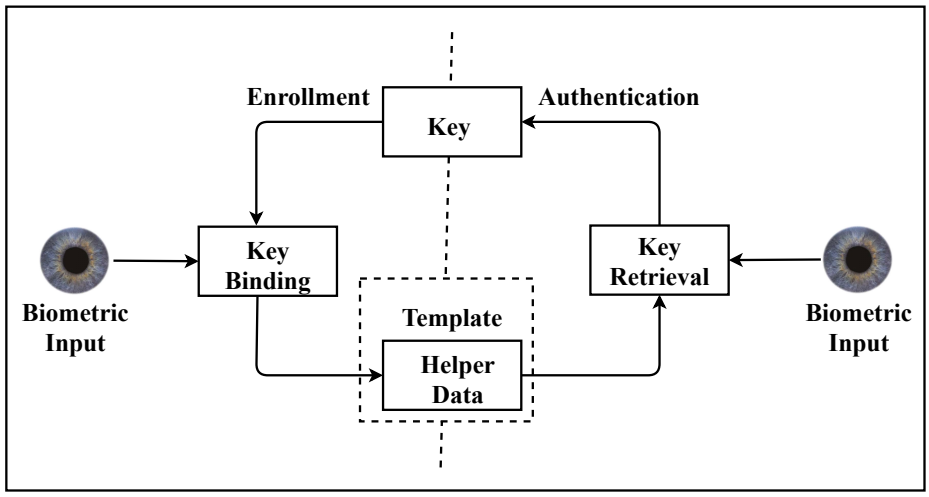


Figure 1.8: Framework of Key Binding Biometric Cryptosystem

1.6.3 Hybrid Methods

A single scheme is not sufficient to satisfy all the requirements of template protection schemes [28]. So, hybrid method scheme is introduced to solve the limitations of using either cancelable biometrics or biometric cryptosystems alone. Hybrid methods of Biometric Template Protection Schemes can be obtained by integrating cancelable biometrics and biometric cryptosystems.

1.6.4 Homomorphic Encryption

Cancelable biometrics suffer from performance degradation [24, 29]. Biometric cryptosystem uses the auxiliary data, compromise of the auxiliary data leads to the leakage of biometric information [30, 31], results in the entire system vulnerability. Homomorphic encryption (HE) is introduced as a BTP scheme to solve the limitations of cancelable biometrics & biometric cryptosystems [32]. Combining HE with BRS would meet the properties of BTP schemes without degradation of the performance. HE is a unique kind of encryption technique which allows operations like multiplication and addition to be performed directly on the encrypted data without accessing the secret key [33].

1.7 Properties, Functions and Categories of Homomorphic Encryption

The advantage of HE is explained with the help of the below scenario.

Scenario: The scenario is shown in Figure 1.9. It consists of two entities, namely Client

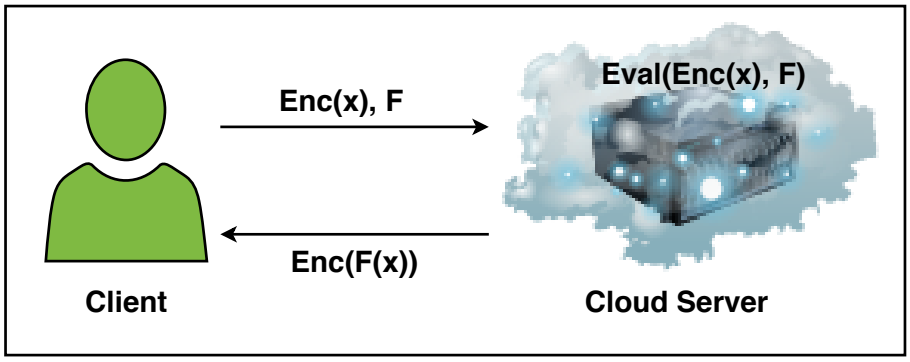


Figure 1.9: Scenario explaining the advantage of homomorphic encryption

and Cloud server. The client owns private data **x** and wants to perform function **F** on **x**. However, the client has very limited computational resources. So, the client wants to outsource the computation to the cloud server. At the same time, the client might not trust the cloud. As a result, instead of sending **x**, the client performs encryption on **x** using homomorphic encryption algorithms like Paillier, ElGamal, etc. and send **Enc(x)** & **F** to

cloud server. Cloud server runs the homomorphic evaluation function. The inputs for the function are **Enc(x)** & **F** and produces the output as **Enc(F(x))** without learning the value of **F(x)**. The server sends **Enc(F(x))** to the client. The client decrypts **Enc(F(x))** to obtain the value of **F(x)**.

1.7.1 Properties of Homomorphic Encryption

Given two encrypted values $Enc(a)$ & $Enc(b)$ for values a, b. The properties of HE are defined as follows:

Additive Property:

It states that the addition of two original values can be obtained by the decryption of multiplication of two encrypted values. The additive property of HE is shown in Figure 1.10 and given in equation (1.1).

$$Dec(Enc(a) * Enc(b)) = a + b \tag{1.1}$$

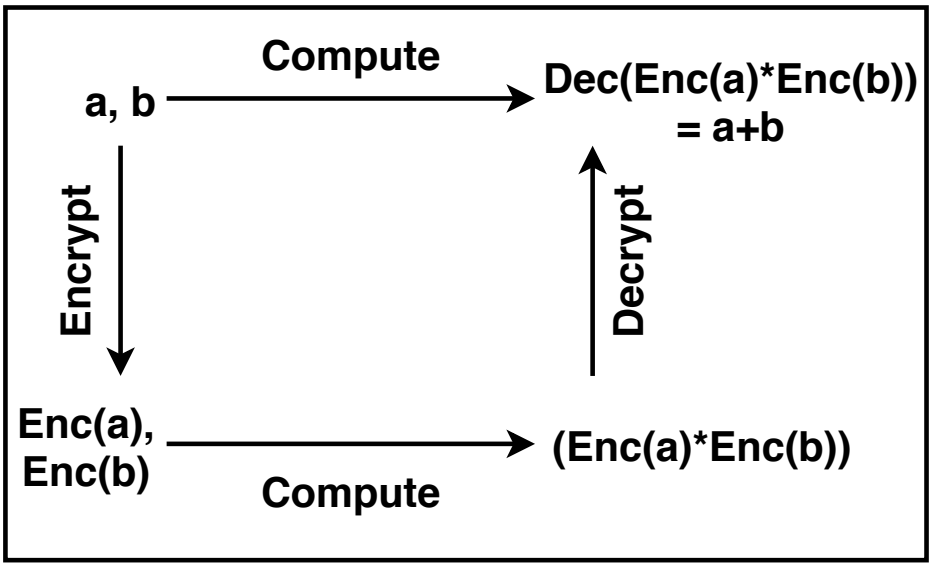


Figure 1.10: Additive Property of homomorphic encryption

Multiplicative Property:

It states that the multiplication of two original values can be obtained by the decryption

of multiplication of two encrypted values. The multiplicative property of HE is shown in Figure 1.11 and given in equation (1.2).

$$Dec(Enc(a) * Enc(b)) = a * b \tag{1.2}$$

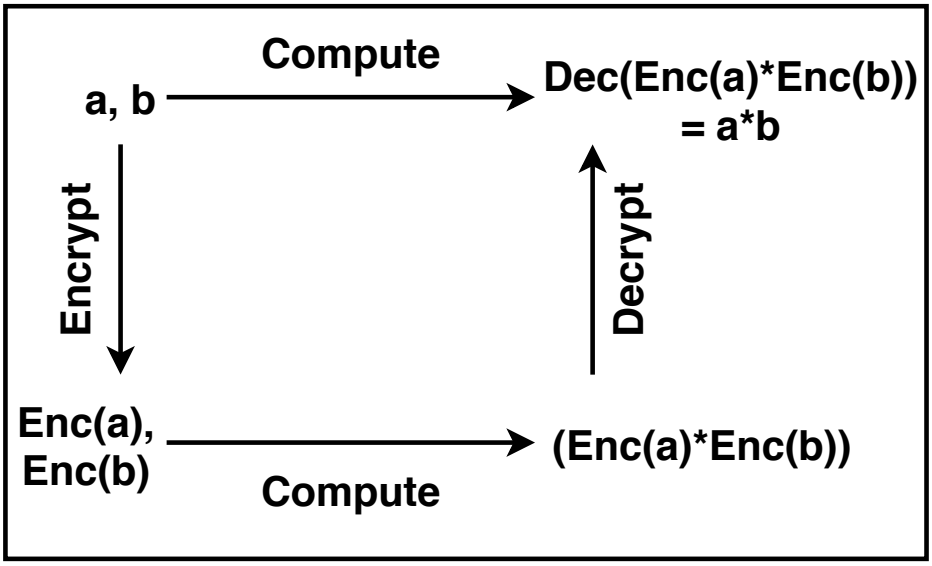


Figure 1.11: Multiplicative Property of homomorphic encryption

1.7.2 Functions of Homomorphic Encryption

HE involves four functions [33, 34], namely Key Generation (KeyGen), Encryption (Enc), Evaluation (Eval), and Decryption (Dec) as shown in Figure 1.12. The details of each function are given below:

1. KeyGen(parameters) $\Rightarrow$ $(P_k,\ S_k)$: The function generates secret (S_k) and public keys (P_k) by using the given security parameters.

2. Enc(P_k, msg) $\Rightarrow$ $\varepsilon(msg)$: For a given P_k and message msg, the function encrypts msg using P_k and outputs a ciphertext $\varepsilon(msg)$.

3. Eval(P_k, C, $\varepsilon(msg_1)$, $\varepsilon(msg_2)$, . . ., $\varepsilon(msg_n)$) $\Rightarrow$ $\varepsilon(R)$: For a given public key P_k, evaluated circuit C, and a group of ciphertexts, $\varepsilon(msg_1)$, $\varepsilon(msg_2)$, . . ., $\varepsilon(msg_n)$, the function outputs a computation result in encrypted form, $\varepsilon(R)$.

4. Dec(S_k, $\varepsilon(R)$) $\Rightarrow$ R: For a given ciphertext $\varepsilon(R)$ and secret key S_k, the function decrypts $\varepsilon(R)$ and produces its original value R.

The evaluation function (Eval) helps to achieve the computation on the cipher texts itself without accessing the secret key.

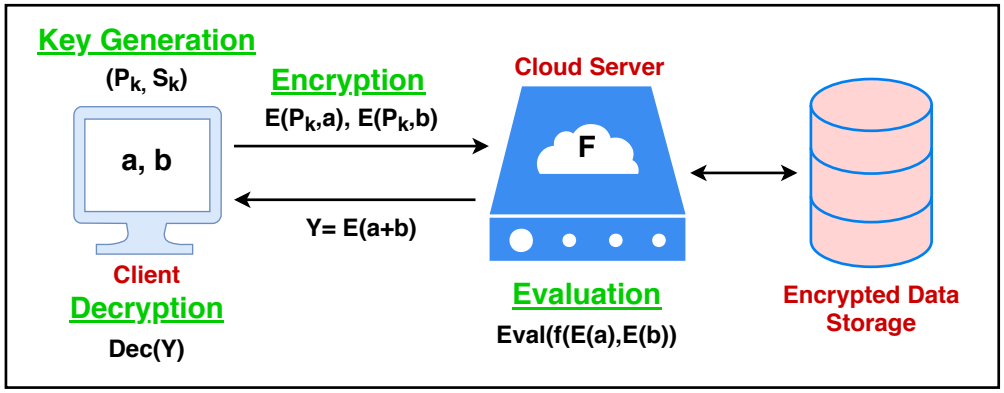


Figure 1.12: Functions involved in homomorphic encryption

1.7.3 Categories of Homomorphic Encryption Schemes

The homomorphic encryption schemes are broadly classified into three types based on the allowed number of operations on encrypted data [33, 34, 35] as shown in Figure 1.5.

1.7.3.1 Partial Homomorphic Encryption (PHE)

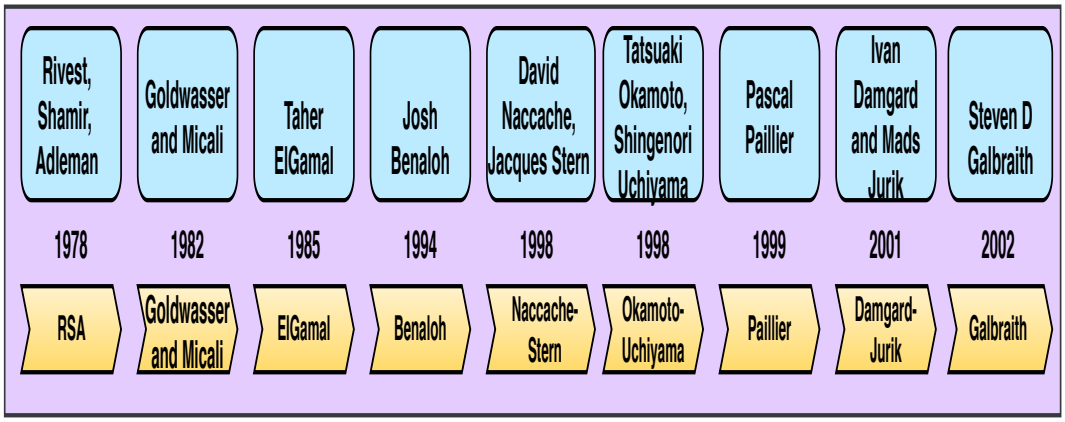


Figure 1.13: Evolution of Partial homomorphic encryption Schemes before Gentry's work [2]

PHE allows to perform either addition or multiplication with an unlimited number of times on encrypted data. The evolution of major PHE schemes is shown in Figure 1.13. In 1978, Rivest *et al.* built a homomorphic scheme named Rivest Shamir Adleman (RSA) [36] which allows multiplication operation on encrypted data. In 1982, Goldwasser & Micali proposed an additive homomorphic scheme named Goldwasser-Micali (GM) [37]. GM allows addition operation on encrypted data. Taher ElGamal improved the Diffe-Hellman key exchange algorithm [38] and developed a scheme named ElGamal [39] in 1985. ElGamal satisfies the multiplicative property of homomorphic encryption. Benaloh enhanced the GM cryptosystem and introduced a homomorphic scheme named Benaloh [40] in 1994 by preserving the additive homomorphic property. GM performs bit-by-bit encryption, whereas Benaloh performs block-wise encryption. Okamoto and Uchiyama built a scheme named Okamoto-Uchiyama (OU) [41] in 1998. OU allows addition operation on encrypted data. In 1999, Paillier proposed an additive homomorphic scheme named Paillier [42]. Naccache and Stern, Damgard and Jurik improved the computational efficiency of Benaloh, Paillier and proposed cryptosystems, namely Naccache & Stern, Damgard & Jurik [43, 44] in 1998, 2001, respectively by preserving the same homomorphic properties. In 2002, Galbraith introduced an additive homomorphic scheme named Galbraith [45] which can be applied on elliptic curves.

1.7.3.2 Somewhat Homomorphic Encryption (SHE)

SHE allows both addition & multiplication but with a limited number of times on encrypted data. The major SHE schemes, which were used as a stepping stone to fully homomorphic encryption, are shown in Figure 1.14. In 1982, Yao built the first SHE scheme [46] where the ciphertext grows at least linearly. Sander *et al.* proposed a SHE scheme named Sander Young Yung (SYY) [47] in 1999 over a semi-group which allows one OR/NOT operation and polynomially many AND operations on encrypted data. In 2005, Boneh *et al.* proposed a SHE scheme named Boneh Goh Nissim (BGN) [48] which allows unlimited addition operations and one multiplication operation. Ishai *et al.* developed a homomorphic encryption technique named Yuval Ishai & Anat Paskin (IP) [49] in 2007 by implementing the branching programs on the ciphertext. Except BGN, the size of ciphertext for Yao, SYY & IP

SHE schemes grows either linearly or exponentially. In BGN, the size of ciphertext grows constantly.

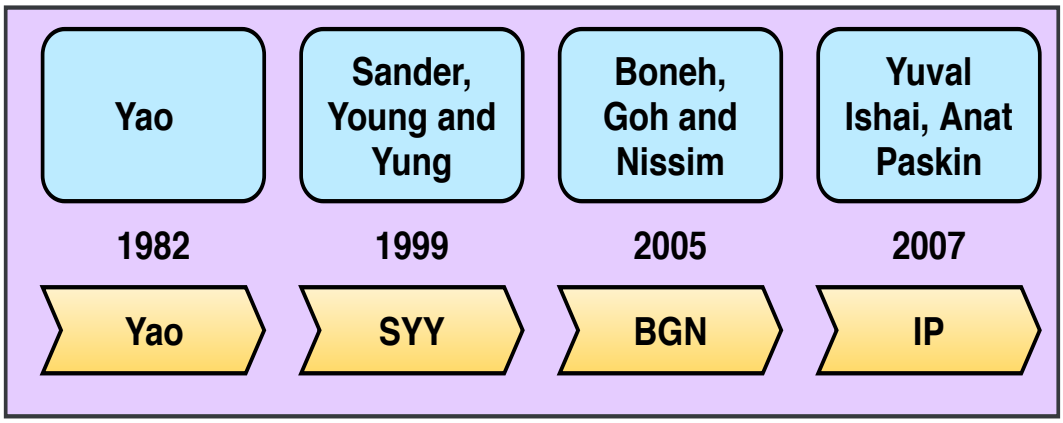


Figure 1.14: Evolution of Somewhat homomorphic encryption Schemes before Gentry's work [2]

1.7.3.3 Fully Homomorphic Encryption (FHE)

FHE allows both addition & multiplication with an unlimited number of times on encrypted data. Gentry made a breakthrough in 2009 and introduced a first FHE scheme [2]. Gentry's scheme is based on ideal lattices and is a framework to obtain an FHE scheme. However, Gentry's scheme is not a practical one. Therefore, a lot of researchers used the framework proposed by Gentry and introduced practically achievable FHE schemes in successive years.

1.8 Information Fusion in Biometrics

The use of several methods or inputs of processing of biometric modalities/samples is known as biometric fusion. Multi-biometric systems (MBS) depend on the evidence presented by various sources of biometric information. The advantages, such as improved efficiency, accuracy, non-universality, less vulnerable to spoofing attack, makes the MBS to be used in various applications over unimodal systems [50]. Single or multiple biometric modalities can be used for biometric fusion and is shown in Figure 1.15

MBS are categorized into six types [51] namely, multi-instance, multi-sample, multi-

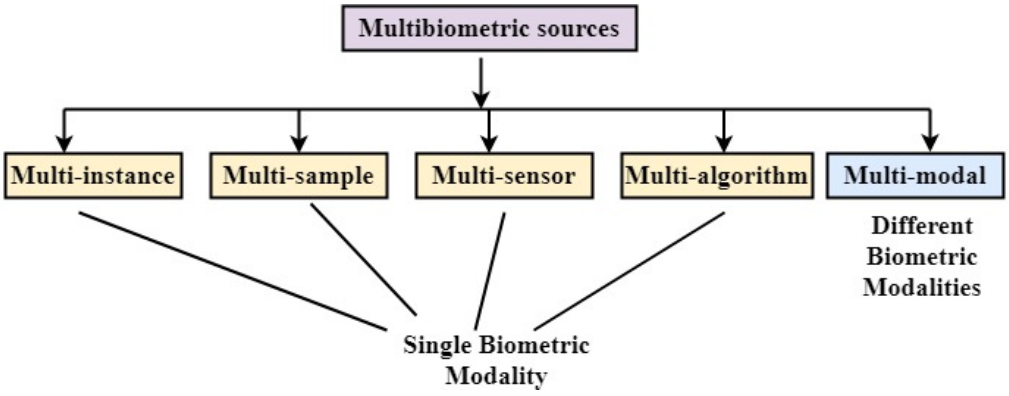


Figure 1.15: Multiple sources of evidence used for fusion

sensor, multi-algorithm, multi-modal & hybrid systems. Single biometric modality is used for fusion in multi-instance, multi-sample, multi-sensor, and multi-algorithm. Multiple modalities are used for fusion in multi-modal systems [52]. Multi-instance systems use multiple instances of the same biometric data (eg. left and right iris of a person). The information collected from several sensors is fused in multi-sensor systems. Several samples of a same biometric modality are collected at different times and fuse in multi-sample systems (eg. left, right and frontal profiles of a face). Different algorithms are used to create feature sets from a single biometric modality, and all the extracted feature sets information are fused in multi-algorithm systems (eg. texture based features and minutiae based features extracted from fingerprint). The information of various biometric modalities is fused in the multi-modal system [51]. Some of the works in this thesis use the multi-instance fusion as they are cost-effective and do not require the additional sensors, need of matching algorithms, and feature extraction methods.

Fusion can be accomplished at various levels, namely sensor level, feature level, decision level or score level [52]. Decision level and score level fusion are considered as fusion after matching and called late fusion [51]. Sensor level and feature level fusion are considered as fusion prior to matching and called early fusion. The feature level fusion provides better recognition rate when compared to other level fusion techniques [53].

1.9 Performance of a Biometric System

The following measures can be used to assess the performance of a biometric system:

- **Genuine Score**: The score computed by matching two samples of a biometric modality belongs to the same user is known as a genuine score.
- **Imposter Score**: The score computed by matching two samples of a biometric modality belongs to different users is known as imposter score.
- **False Accept Rate (FAR) or False Match Rate (FMR)**: The ratio of imposter scores exceeding the threshold to the total imposter scores is known as the FMR.

$$FMR = \frac{Number\ of\ Imposter\ scores\ exceeding\ the\ threshold}{Total\ imposter\ scores} \tag{1.3}$$

- **False Reject Rate (FRR) or False Non-Match Rate (FNMR)**: The ratio of genuine scores exceeding the threshold to the total genuine scores is known as the FNMR.

$$FNMR = \frac{Number\ of\ Genuine\ scores\ exceeding\ the\ threshold}{Total\ Genuine\ scores} \tag{1.4}$$

- **Genuine Accept Rate (GAR)**: GAR is defined as the percentage of genuine users accepted by the system. Therefore, GAR =l - FRR.
- The Receiver Operating Characteristic (ROC) or Detection Error Tradeoff (DET) curve is used to measure the efficiency of a biometric system. ROC curve is plotted by taking FAR on X-axis and GAR on Y-axis. In ROC curve, linear, or semi-logarithmic scale is used whereas a logarithmic scale is used in DET curve.
- The point at which the FAR equals the FRR is referred as Equal Error Rate (EER). The better performance is indicated by a lower EER value.
- The d-prime value (d') and Kolmogorov-Smirnov (KS) - test are used to measure

how well the genuine and imposter scores are separated. The d' is given by

$$d' = \frac{|m_{genuine} - m_{imposter}|}{\sqrt{(sd^2_{genuine} + sd^2_{imposter})/2}} \tag{1.5}$$

where sd, and m indicate standard deviation and mean of imposter and genuine distributions. The better performance is indicated, by the larger d' value. The range of KS-test value is [0, 1]. KS-test value closer to 1 indicates more separation between the imposter and genuine scores.

1.10 Benchmark Databases

The following databases are considered to validate the efficiency of the system.

- CASIA-V 1.0 [54] contains 108 subjects. Each subject consists of 7 samples.
- CASIA-V3-Interval [55] contains 172 subjects of the left eye & 165 subjects of the right eye. Each subject consists of 5 samples.
- IIT-Delhi (IITD) [56] contains 224 subjects. Each subject consists of 5 left & right samples.
- SDUMLA-HMT [57] contains 106 subjects. Each subject consists of 5 left & right samples.

The University of Salzburg tool kit [58] is used to extract the iris code from the iris images in the databases. The first five samples from each subject are considered to perform the experiments. Subjects consisting of minimum 5 left & right samples are required to develop a multi-instance iris recognition system. So, 106, 208 & 115 subjects from SDUMLA-HMT, IITD & CASIA-V3-Interval iris databases are considered to check the efficiency of Blockchain-based multi-instance iris authentication system, Secure & verifiable multi-instance iris authentication system using public auditor and Secure & verifiable multi-instance iris authentication system using Blockchain as the subjects contain both left & right irises with a minimum of 5 samples each.

1.11 Motivation for present work, Aim & Objectives

The exposure of biometric modalities in a variety of applications for verification makes a serious compromise on user's privacy [6]. To address this, biometric templates are protected using BTP schemes such as cancelable biometrics, biometric cryptosystems and homomorphic encryption [29, 35]. Homomorphic encryption is the most recent explored research area to construct privacy-preserving biometric authentication systems due to its advantages over cancelable biometrics and biometric cryptosystems [32]. But studies in the existing literature assumed that the server is honest-but-curious. In a malicious server setting, the server may return an arbitrary result to save the computational resources results in false accept or false reject.

1.11.1 Aim

This dissertation aims to provide secure & verifiable methods for iris authentication on a malicious cloud server by maintaining the trade-off between accuracy & security.

1.11.2 Objectives

The main objectives of this dissertation are stated as follows:

- To get insight into the state-of-the-art privacy-preserving biometric authentication system.
- To understand the existing template protection methods using homomorphic encryption and provide solutions for better security & performance.
- To use advanced technology such as Blockchain in BRS to solve the override comparator attack.
- To implement machine learning classification techniques on encrypted data and use these techniques for secure authentication.
- To propose a fusion technique which maximizes the pair-wise correlations and minimizes the between-class correlations.

- To understand the research gap that needs to be addressed and to find the future directions in the field of BRS.

1.12 Overview of the Contributions of the Thesis

This thesis provides the following contributions made for secure & verifiable methods for iris authentication on a untrusted server.

1. Proposed a privacy preserving iris authentication technique using Fan-Vercauteren scheme, which generates rotation-invariant iris template yields higher recognition accuracy and perform hamming distance computation between encrypted reference and probe template results in preserving the privacy of user's data.

2. Proposed a Blockchain-based multi-instance iris authentication system (BMIAE), which integrates ElGamal homomorphic encryption [39] with Blockchain technology to achieve privacy of iris templates and trust on the comparator result. The challenges of using Blockchain in biometrics are also addressed in BMIAE.

3. Proposed a method for multi-instance iris authentication on a malicious cloud server (SviaPA), which not only provides privacy for the iris templates but also includes a verification procedure to check whether the comparator result is correct or not.

4. A method for secure and verifiable iris authentication using Blockchain (SviaB) is proposed. SviaB combines Blockchain technology with Paillier homomorphic encryption [42]. Paillier homomorphic encryption provides confidentiality for the iris templates. The Blockchain provides the integrity of the encrypted reference iris templates as well as the trust of the comparator result. In addition, SviaB reduces the time taken to authenticate a person when compared to BMIAE.

5. A secure and verifiable machine learning-based iris authentication method (SvaS) is proposed. SvaS performs both privacy-preserving (PP) training & classification phases on the encrypted data. The public verifier can verify the correctness of the classification result computed by the cloud server using a verification procedure. The

nearest neighbor & multi-class perceptron classification algorithms are implemented on encrypted data.

6. Proposed a feature level fusion technique, namely Contradistinguish Similarity Analysis (CSA) which increases the correlations between samples of different class and reduces the correlations between samples of the same class. It also includes a verification procedure by using polynomial factorization algorithm to verify the result returned by the cloud server.

1.13 Thesis Organization

The rest of the chapters of this thesis are organized as follows: Chapter 2 describes the recent state-of-the-art works on homomorphic encryption applied to biometric recognition, machine learning on encrypted data, machine learning approaches applied to iris recognition and Blockchain for biometrics. This chapter reports the extraction outcomes resulted from the analysis of literature.

Chapter 3 assumes that the server is Honest-but-curious and presents a privacy-preserving iris authentication system to solve the limitations of cancelable biometrics and biometric cryptosystems. It also provides a solution to solve the rotational inconsistency problem due to head tilt of a person during the authentication phase. Chapter 4 presents three multi-instance iris authentication systems which not only provide the confidentiality of the iris templates but also trust on the matching result. The advantage of using emerging technology like Blockchain in biometrics is explored here.

Chapter 5 assumes that the server is a malicious entity and presents two secure and verifiable iris authentication systems by using private machine learning classification. It also presents a technique, contradistinguish similarity analysis (CSA) for effective feature level fusion. The conclusions of the thesis and future directions are outlined in chapter 6. The techniques in chapter 4 & 5 are trustworthy against a malicious server and eliminates the need to trust any third-party or a server for comparator result. All the techniques in chapter 3, 4, 5 satisfy all the properties of biometric template protection schemes.

Chapter 2

Literature Survey

In this chapter, a brief survey of the literature related to the contributions made in this thesis is given. The chapter is organized as follows: Section 2.1 covers some studies related to homomorphic encryption schemes applied to biometric recognition. Works related to machine learning techniques applied to iris recognition are given in Section 2.2. Section 2.3 discusses some studies related to machine learning techniques applied on encrypted data. Works related to applying Blockchain technology for Biometrics are discussed in Section 2.4. The publicly available implementations of some FHE schemes, research communities working on template protection schemes, publicly available databases and its sources are listed in Section 2.5. Finally, the summary of this chapter is provided in Section 2.6.

2.1 Homomorphic Encryption applied to Biometric Authentication

Upmanyu *et al.* [59] suggested a secure protocol for biometric verification named "Blind Authentication" by using Rivest Shamir Adleman (RSA) and Paillier [42]. Blind Authentication protocol considered the enrollment server is a trusted entity; as a result, it provides only privacy-preserving (PP) classification and fails to provide PP enrollment. Osadchy *et al.* [60] proposed a secure face identification ("Scifi") system by using Paillier cryptosystem [42]. The Scifi system yields superior results when compared to the existing works,

even in illumination invariant conditions. Rahulamathavan *et al.* [61] suggested a method to recognize the expression of a face by using the properties of Paillier and computed the required operations on encrypted data. Pastoriza *et al.* [62] introduced a secure face verification system in a non-interactive manner which can be applied to lightweight devices. The authors proposed a homomorphic encryption scheme to accomplish the matching on encrypted data. Sedenka *et al.* [63] designed a secure biometric authentication in an outsourced environment. Penn *et al.* [64] used the Paillier homomorphic scheme [42] and proposed a biometric matching technique which performs better than Goldwasser-Micali approach. Authors applied the matching technique on iris biometric to validate the efficiency. Haghighat *et al.* [65] suggested a biometric verification in a cloud environment. The method uses a searching-based matching instead of distance-based matching. Yasuda *et al.* [66] proposed two packing techniques to reduce the size of the encrypted data results in better performance. These techniques are applied for secure biometric authentication.

Xiang *et al.* [67] introduced a secure face recognition with computation in a cloud server by using public key encryption & fully homomorphic encryption algorithm. The client is able to validate the result computed by the cloud server. Hahn *et al.* [68] introduced an secure & efficient identification system by using symmetric homomorphic encryption. The system performs better when compared to the existing works. Gomez *et al.* [32] proposed a template protection approach for multi-biometric recognition using Paillier. The final comparison is performed on the plaintext by the server; as a result, introduces a breach into the security of the system. Santosh *et al.* [69] used the Paillier and Elliptic curve encryption techniques to provide privacy of biometric templates which are stored in a cloud server. Taheri *et al.* [70] suggested a method on encrypted data using correlation filters and homomorphic scheme. The privacy of the iris templates is achieved by storing only the each class correlation filter instead of templates. Naresh *et al.* [71] presented an approach to secure the database of face templates and to perform matching on the encrypted face templates by using fan-vercauteran scheme [72].

Zhu *et al.* [73] designed a method named efficient fingerprint authentication (“e-Finga”)

Article's reference	Biometric Modality	HE scheme	Database	Performance Measures	Result
Upmanyu *et al.* [59]	Iris, Fingerprint, Hand Geometry, Face	RSA & Paillier	Casia-V 1.0, FVC2004 DB2 In-house, Yale	Accuracy	Fingerprint: 84.45%, Iris: 98.24% Face: 96.91%, Hand Geometry: 98.38%
Osadchy *et al.* [60]	Face	Paillier	FERET CMU-PIE	Accuracy and time	CMU-PIE: 91.1% FERET: 92% 31 seconds
Rahulamathavan *et al.* [61]	Face	Paillier	JAFFE and MUG	Accuracy	JAFFE: 94.37%, MUG: 95.2%
Pastoriza *et al.* [62]	Face	GH11	XM2VTS, FERET, LFW	Accuracy	XM2VTS: 98.37%, FERET: 97.77% LFW: 69.53%
Sedenka *et al.* [63]	Touch screen behavior	Damgard, Geisler and Kroigaard	Dataset-LTU, Dataset-Frank	EER	Dataset-LTU: 0.231, Dataset-Frank: 0.183
Penn *et al.* [64]	Iris	Paillier	CASIA-V3-Interval	Time	3 seconds
Haghighat *et al.* [65]	Face	Boneh and Waters Searchable Encryption	FERET	Accuracy Time	95.00% 18 seconds
Yasuda *et al.* [66]	Any	Polynomial-LWE	-	Time	5.31 milliseconds
Xiang *et al.* [67]	Face	Paillier	-	-	-
Hahn *et al.* [68]	Fingerprint	Symmetric HE	-	Time	15 seconds and 2 minutes for 256MB and 4GB databases
Gomez *et al.* [32]	Fingerprint + Signature	Paillier	Biosecure ID	EER	Feature level Fusion: 0.12% (Euclidean), 3.00% (Cosine) Score level Fusion: 0.74% (Euclidean), 1.25% (Cosine) Decision-level Fusion: 1.19% (Euclidean), 1.71% (Cosine)
Santosh *et al.* [69]	Face	Paillier + Elliptic curve	FERET	Accuracy Time	96.89% 345 seconds
Taheri *et al.* [70]	Face	Paillier	LFW Yale-B FERET	EER	LFW: 4.84% Yale: 2% FERET: 1.09%
Naresh *et al.* [71]	Face	Fan-Vercauteren	LFW IJB-A, B CASIA	Accuracy	LFW: 98.72% IJB-A: 73.66% IJB-B: 74.66% CASIA: 93.33%
Zhu *et al.* [73]	Fingerprint	2DNF	FVC2006 DB1	Time	1.5 seconds
Zhou *et al.* [74]	Fingerprint	Threshold Predicate Encryption	-	Time	1 second for 2000-bit template
Lee *et al.* [75]	Any	Single-key function hiding Inner product encryption	-	Time	Client: 3.12 milliseconds Server: 0.0021 milliseconds
Barni *et al.* [76]	Iris + Face	Speedz Protocol	CASIA-V 1.0 CASIA-Face V5	EER Time	0.98% 0.05 seconds
Guo *et al.* [77]	Face	Randomness techniques	ORL	Number of operations	m Multiplications, m is the number of faces
Topcu *et al.* [78]	Fingerprint	-	FVC2002 DB1A FVC2002 DB2A	EER	0.995% 0.907%

Table 2.1: Summary of selected works under Homomorphic encryption schemes applied to biometric recognition.

for secure online fingerprint authentication. e-Finga uses lightweight multi-party polynomial aggregation & multi-party random masking techniques to provide security. A light weighted encryption scheme named "Threshold Predicate Encryption (TPE)" is proposed by Zhou *et al.* [74]. A PP user-centric authentication system named "PassBio" is proposed by using TPE. Lee *et al.* solved the limitations of PassBio in [75] by using single-key function-hiding inner product encryption. Hu *et al.* [79] suggested single-server and two-server solutions to preserve the privacy of iris templates by performing the computations on the encrypted templates. Single-server solution uses the symmetric key algorithm and two-server solution uses somewhat homomomorphic encryption scheme. Barni *et al.* [76] designed a secure multi-modal biometric authentication ("SEMBA"), which combines iris and face templates. Guo *et al.* [77] use randomness techniques instead of homomorphic encryption to provide the privacy of the face templates result in a good performance. Topcu *et al.* [78] proposed a framework for secure fingerprint authentication system. The authors generated the fixed-length binary templates by leaving the security as future work. The summary of homomorphic encryption schemes applied to biometric recognition is shown in Table 2.1.

2.2 Machine Learning approaches applied to Iris Recognition

Sibai *et al.* [95] designed an iris recognition system by using feed forward artificial neural network. Authors conducted several experiments by varying the input format, number of hidden layers, and the number of neurons in the hidden layer to find the optimal parameters. Khedkar and Ladhake [80] proposed an iris recognition system using neural network techniques such as support vector machines (SVM), radial basis function (RBF) and multi-layer perceptron (MLP). Rai *et al.* [81] suggested a method to identify the iris patterns by using SVM and Hamming distance. Authors proposed two feature extraction techniques, namely 1D Log Gabor wavelet and Haar wavelet decomposition. Srivastava *et al.* [82] implemented an approach for iris recognition by combining functional modular neural net-

Article's reference	Machine learning technique	Database	Performance Measures	Results
Khedkar *et al.* [80]	MLP, RBF, SVM	CASIA-V 1.0	Accuracy	95%
Rai *et al.* [81]	SVM	CASIA-V 1.0, Chek	Accuracy	99.91%, 99.88%
Srivastava *et al.* [82]	Functional modular neural networks + Evolutionary fuzzy clustering	CASIA-V 1.0	Accuracy	98.12%
Saminathan *et al.* [83]	Kernal-based multi -class SVM	CASIA-V 1.0	Accuracy	99.3%
Ahmadi *et al.* [84]	MLP and PSO	CASIA-V3-Interval	Accuracy	95.36%
Fahim *et al.* [85]	SVM, KNN Linear Discriminant Analysis	Trokielewicz [86]	Accuracy	97%
Ahmadi *et al.* [87]	MLP-ICA	CASIA-V3-Interval	Accuracy	99.99%
Waisy *et al.* [88]	CNN Soft-max classifier	CASIA-V3-Interval IITD SDUMLA-HMT	Accuracy	100% 100%
Ahmadi *et al.* [89]	Hybrid radial basis function neural network with genetic algorithm	CASIA-V3-Interval UBIRIS.V1	Accuracy	99.99% 99.98%
Arsalan *et al.* [90]	Fully residual encoder-decoder network	CASIA-V4.0 Interval IITD UBIRIS V2.0	Accuracy	99.10% 98.41% 98.52%
Zhao and Ajay [91]	mask R-CNN Fully CNN ETL function	CASIA-V4.0 Interval IITD ICE 2006 WVU non-ideal	EER	4.07% 0.68% 1.12% 2.20%
Wang and Ajay [92]	CNN + supervised discrete hashing	PolyU B1 Cross-spectral	EER	5.31% 6.34%
Zhao *et al.* [93]	Deep CNN + Capsule network	JluIrisV 3.1 JluIrisV 4	Accuracy	99.37% 99.42%
Adamovic *et al.* [94]	Random forest	CASIA-V4.0 Interval IITD MMU	Accuracy	99.99%
Sibai *et al.* [95]	Feed forward neural network	IITD	Accuracy	93.33%
Gale *et al.* [96]	weighted DAG SVM + SNN	CASIA-V 1.0	Accuracy	99.99%

Table 2.2: Summary of selected works under machine learning techniques applied to iris recognition.

works and evolutionary fuzzy clustering. Saminathan *et al.* [83] introduced a method for iris authentication by using kernel-based multi-class SVM. Marsico *et al.* [97] presented a survey of machine learning techniques ranging from neural networks to deep learning for iris recognition. An iris recognition system is proposed by Ahmadi *et al.* [84] to increase generalization performance by using particle swarm optimization and MLP. The authors extended their work in [89] by using RBF with a genetic algorithm to reduce the computational complexity. Fahim *et al.* [85] proved the feasibility of machine learning techniques to recognize a person with iris modality even if an eye image is captured through a smartphone.

Ahmadi *et al.* [87] designed an iris recognition system by using MLP-imperialist competitive algorithm (MLP-ICA) as a classifier. The authors used Gray-level difference matrix to extract the features from the iris. The convolutional neural network (CNN) and softmax classifier are used to extract the features from the iris image and classify the user into any of the N classes by Waisy *et al.* [88]. The method performs better when compared to existing approaches. A deep learning model is designed by Arsalan *et al.* [90], which determines the true iris region without pre-processing the eye image. Unlike existing approaches, the performance is not affected by non-ideal situations. Zhao and Ajay [91] used fully convolutional network and proposed a framework for accurate iris detection, segmentation and recognition. Authors developed an "Extended Triplet Loss (ETL)" function to learn the spatially corresponding features of an iris image. A cross-spectral iris recognition system is designed by Wang *et al.* [92]. The features are extracted by using CNN and supervised discrete hashing (SDH) is used for compression and classification. Admovic *et al.* [94] proposed an approach for iris recogntion by using stylometric features and random forect machine learning methods. The hybrid based particle swarm optimization (PSO) is used as a classifier and proposed an iris recognition system by Gale *et al.* [96]. Hybrid based PSO is a combination of weighted directed acyclic graph (DAG) SVM and spiking neural networks (SNN). The classification task is achieved by weighted DAG SVM and evaluation is achieved by SNN. The summary of machine learning techniques applied to iris recognition is shown in Table 2.2.

2.3 Machine Learning on Encrypted Data

The research on machine learning on encrypted data is broadly classified into two types: Privacy-preserving training and Privacy-preserving classification. Privacy-preserving training refers to building the machine learning model using the encrypted training data. In privacy-preserving classification works, the researchers assume that the model was already build on unencrypted data and model parameters are stored in an encrypted form. During the classification phase, the test instance is encrypted and classified using the model parameters.

Orlandi *et al.* [98] used Paillier cryptosystem [42] to achieve the neural network-based privacy-preserving computation. The protocol achieved privacy-preserving classification but failed to achieve privacy-preserving training. Barni *et al.* [99] proposed two classifiers, namely linear branching programs and neural networks in a privacy-preserving manner to classify the electrocardiogram (ECG) signals. The classifiers are build by using the Paillier cryptosystem [42] and Garbled circuits. Graphel *et al.* [100] used somewhat homomorphic encryption scheme to train linear means classifier and fisher's linear discriminant classifier. In the proposed method, the authors concentrated on privacy-preserving training rather than privacy-preserving classification. There is a leakage of information about the model apart from the result of the classification. The privacy-preserving clinical decision support systems are designed by Rahulamathavan *et al.* [101] and Zhu *et al.* [102]. The former used Gaussian Kernel-based SVM and the latter used Non-linear SVM to diagnose the patient's disease in a secure manner.

Liu *et al.* [103] suggested a privacy-preserving patient-centric clinical decision system using naive Bayesian classifier. In their proposed method, the old patient data is encrypted by using Paillier cryptosystem [42] and the encrypted data is used to train the naive Bayesian classifier. The secure trained classifier is used to predict the disease risk for the new patient. Two multi-key secure deep-learning schemes are proposed by Li *et al.* [104] to minimize the communication and computational cost. Later the authors proposed a secure classifica-

tion framework [105] in an outsourced environment. The scheme generates different public keys to retain the secrecy of client and data provider. The classification protocols such as private decision tree classification, private hyperplane decision based classification, and private naive Bayes classification are constructed by Bost *et al.* [106] using Paillier cryptosystem [42] and Quadratic Residuosity. The number of interactions between the server and the client required to implement the protocols is 2. The number of interactions are reduced from 2 to 1 by Sun *et al.* [107] using an improved version of Fan-Vercauteran scheme [72].

The privacy of back propagation neural network (BPNN) learning algorithm is presented in Yuan *et al.* [109] which reduces the communication and computation costs of each party. Zhang *et al.* [110] utilized the Brakerski, Gentry and Vaikuntanathan (BGV) [111] to preserve the privacy of BPNN learning algorithm. Bachrach *et al.* [113] implemented the neural networks on encrypted data (CryptoNets) by using the properties of homomorphic encryption. The efficiency of CryptoNets for deeper neural networks is improved by Chabanne *et al.* [118] with the help of batch normalization principle. Li *et al.* [115] used the additive property of Paillier [42] and implemented the classification phases of naive Bayes and hyperplane decision-based classifiers in a privacy-preserving manner. Abadi *et al.* [112] utilized the differential privacy and proposed a secure deep learning scheme in an outsourced environment. The summary of machine learning techniques applied to iris recognition is shown in Table 2.3.

2.4 Blockchain for Biometrics

Delgado-Mohatar *et al.* [119] presented the advantages and limitations of using blockchain in biometrics and vice versa. The authors extended their work in [120] to store the biometric templates in the blockchain by using on-chain, direct hashing and Merkle-trees. The storage cost and execution time are less for Merkle-tree based storage when compared to on-chain and direct hashing. The limitations of blockchain for biometrics is not addressed. Delgado-Mohatar *et al.* [121] also analyzed the cost and performance factors to store the

Article's reference	**Security scheme**	**Machine Learning technique**	**PP Training**	**PP Classification**
Orlandi *et al.* [98]	Paillier [42]	Neural Network	No	Yes
Barni *et al.* [99]	Paillier [42] and Garbled Circuits	Linear Branching Program Neural Network	No	Yes
Graphel *et al.* [100]	SHE [108]	Linear Means Fisher's linear discriminant	Yes	No
Rahulamathavan *et al.* [101]	Paillier [42]	SVM	No	Yes
Yuan *et al.* [109]	Boneh, Goh & Nissim doubly homomorphic [48]	Neural Network	Yes	No
Bost *et al.* [106]	Paillier [42] Quadratic Residuosity	Decision tree Hyperplane decision-based Naive Bayes	No	Yes
Liu *et al.* [103]	Paillier [42]	Naive Bayes	Yes	Yes
Zhang *et al.* [110]	BGV scheme [111]	Neural network	Yes	No
Zhu *et al.* [102]	Random Masking Polynomial aggregation	Nonlinear Kernel SVM	No	Yes
Abadi *et al.* [112]	Differential Privacy [111]	Deep learning	Yes	No
Bachrach *et al.* [113]	FHE	Neural network	Yes	No
Li *et al.* [104]	Multi-key FHE [114]	Multi-layer Neural Network	Yes	Yes
Sun *et al.* [107]	Fan & Vercauteren [72]	Decision tree Hyperplane decision-based Naive Bayes	No	Yes
Li *et al.* [115]	FHE	Hyperplane decision-based Naive Bayes	No	Yes
Wang *et al.* [116]	Quadratic Residuosity of Goldwasser Micalli [117]	Decision tree	No	Yes

Table 2.3: Summary of selected works under machine learning classification techniques applied on encrypted data.

protected and unprotected biometric templates and on-chain, off-chain biometric matching. Mohsin *et al.* [122] used blockchain to achieve the integrity and availability in finger-vein verification system.

S.No	Community	Accessible at
1	Biometrics Research Group, Michigan Sate University, USA	http://biometrics.cse.msu.edu/
2	da/sec Biometrics and Internet Security, Research Group, Center for Advanced Security Research Darmstadt (CASED), Germany	https://www.dasec.h-da.de/
3	The Multimedia Signal Processing and Security Lab(WaveLab), University of Salzburg, Austria	http://wavelab.at/member-uhl.shtml
4	IBM Research, Thomas J. Watson Research Center, USA	https://www.research.ibm.com/labs/watson/
5	Multimedia Security Lab, Yonsei University, South Korea	https://sites.google.com/site/multimediasecuritylab/
6	Yokohama Research Laboratory, Hitachi Ltd, Japan	http://www.hitachi.com/rd/about/
7	Advanced Cryptosystems Research Group, National Institute of Advanced Industrial Science and Technology (AIST), Japan	https://www.aist.go.jp/aist_e/list/highlights/2015/vol4/index.html
8	La Trobe University, Australia	http://www.latrobe.edu.au/
9	University of New South Wales at the Australian Defence Force Academy(UNSW@ADFA), Australia	https://www.unsw.adfa.edu.au/
10	Centre for Automation Research, University of Maryland, USA	http://www.cfar.umd.edu/
11	Biometric Systems and Multimedia Forensics LAB, University of "Roma TRE", Italy	http://biomedia4n6.uniroma3.it/index.html
12	Biometrics Systems Laboratory, University of Bolgona, Italy	http://biolab.csr.unibo.it/home.asp
13	CyLab, Biometrics Center, Carnegie Mellon University, USA	https://www.cylab.cmu.edu/research/biometrics.html
14	Universiti tunku abdul rahman Kuala Lumpur, Malaysia	www.utar.edu.my

Table 2.4: Active Research Communities for Biometric Template Protection schemes

S.No	Scheme	Open source library name	Language used	Libraries	Available at
1	Smart & Vercauteren [123]	libScarab [124]	C	GMP, FLINT MPFR, MPIR	https://github.com/hcrypt-project/libScarab
2	Brakerski *et al.* [111]	HElib [125]	C++	GMP, NTL	https://github.com/homenc/HElib
3	Ducas & Micciancio [126]	FHEW	C++	FFTW	https://github.com/lducas/FHEW
4	Chilloti *et al.* [127]	TFHE	C++	FFTW	https://github.com/tfhe/tfhe
5	Fan & Vercauteren [72]	SEAL [128]	C++	No external & dependency	https://github.com/microsoft/SEAL
6	Cheon *et al.*[129]	HEAAN [129]	C++	GMP, NTL	https://github.com/kimandrik/HEAAN
7	Dai *et al.* [130]	cuHE [130]	C++	GMP, NTL	https://github.com/vernamlab/cuHE
8	Rohloff [131]	PALISADE [131]	C++	GMP, NTL	https://git.njit.edu/palisade/PALISADE

Table 2.5: Some open-source FHE implementations.

S.No	Database	Biometric trait	Abbreviation	Source
1	CASIA Iris version 1	Iris	National Laboratory of Pattern Recognition Institute of Automation, Chinese Academy of Sciences	http://biometrics.idealtest.org/ dbDetailForUser.do?id=1
2	CASIA Iris version 3			http://biometrics.idealtest.org/ dbDetailForUser.do?id=3
3	CASIA Iris version 4			http://biometrics.idealtest.org/ dbDetailForUser.do?id=4
4	ICE		Iris Challege Evaluation	http://www.nist.gov/itl/ iad/ig/ice.cfm
5	IITD iris v1		IIT Dellhi Iris version 1	http://www4.comp.polyu.edu.hk/ csajaykr/IITD/Database_Iris.htm
6	FVC 2000	Fingerprint	Fingerprint Verification Competetion	http://bias.csr.unibo.it/fvc2000/
7	FVC 2002			http://bias.csr.unibo.it/fvc2002/
8	FVC 2004			http://bias.csr.unibo.it/fvc2004/
9	FVC 2006			http://bias.csr.unibo.it/fvc2006/
10	NIST-SD14		National Institute of Standards and Technology- Special Database	http://www.nist.gov/srd/ nistsd14.cfm
11	FERET	Face	Facial Recognition Technology	http://www.nist.gov/itl/iad/ ig/colorferet.cfm
12	CALTECH		California Institute of Technology.	http://www.vision.caltech.edu/ html-files/archive.html
13	CMU-PIE		Carnegie Mellon University- Pose, Illumination and Expression	http://vasc.ri.cmu.edu/idb/ html/face/
14	NIR face		Hong Kong Polytechnic University, Near-Infrared	http://www4.comp.polyu.edu.hk/ biometrics/polyudb_face.htm
15	ORL face		AT& T Laboratories, Cambridge	http://www.cl.cam.ac.uk/ research/dtg/attarchive/ facedatabase.html
16	NVIE		Natural Visible and Infrared Facial Expression	http://nvie.ustc.edu.cn/
17	FRGC		Face Recognition Grand Challenge	http://www.nist.gov/ itl/iad/ig/frgc.cfm
17	XM2VTS		Multi Modal Verification for Teleservices and Security applications	http://www.ee.surrey.ac.uk/ CVSSP/xm2vtsdb/
18	AR face		-	http://www2.ece.ohio-state.edu/ aleix/ARdatabase.html
19	Poly U	Palmprint	Hong Kong Polytechnic University Palmprint	http://www4.comp.polyu.edu.hk/ biometrics/ MultispectralPalmprint/MSP.htm
20	SVC 2004	Signature	Signature Verification Competetion	http://www.cse.ust.hk/svc2004/
21	MCYT		(Ministerio de Cienciay Tecnologıa, Spanish Ministry of Science and Technology	MCYT baseline corpus [132]
22	TI 46	Voice	Texas Instruments 46-Word Speaker-Dependent Isolated Word Corpus	http://catalog.ldc.upenn.edu/ LDC93S9
23	YOHO speech		YOHO Speaker Verification	https://catalog.ldc.upenn.edu/ LDC93S9

Table 2.6: Publicly available databases on which the methods in the literature are evaluated and their source.

2.5 Resources and Analysis

The research communities who are working to protect the biometric templates by using various BTP schemes are mentioned in Table 2.4. Some of the FHE implementations are made available as an open-source by several researchers are listed in Table 2.5. In this thesis, Simple Encrypted Arithmetic Library (SEAL) implemented by Cheon *et al.* [128] is used to perform the operations on encrypted data. The publicly available databases on which the homomorphic encryption schemes are applied to protect the biometric templates discussed in the literature are evaluated along with their source are listed in Table 2.6.

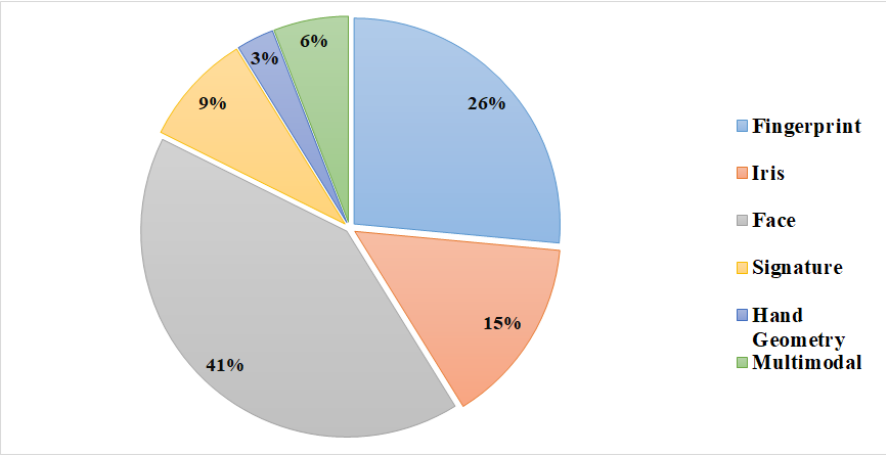


Figure 2.1: Percentage of Homomorphic Encryption schemes applied to each biometric trait

The percentage distribution of HE schemes applied to each biometric trait is illustrated in Figure. 2.1. We can infer from the Figure 2.1 that BTP schemes based on HE developed so far are 41% on the face, 21% on the fingerprint, 15% on the iris, 9% on the signature, 6% on multi-modal and 3% on Hand Geometry. The percentage distribution of PHE, SHE and FHE applied to biometric recognition is shown in Figure. 2.2. It is observed that 67%, 20% and 13% works are used PHE, SHE and FHE schemes respectively. Hence, we can say that FHE schemes need to be applied to biometric recognition as a BTP scheme to make use of the advantages of FHE schemes.

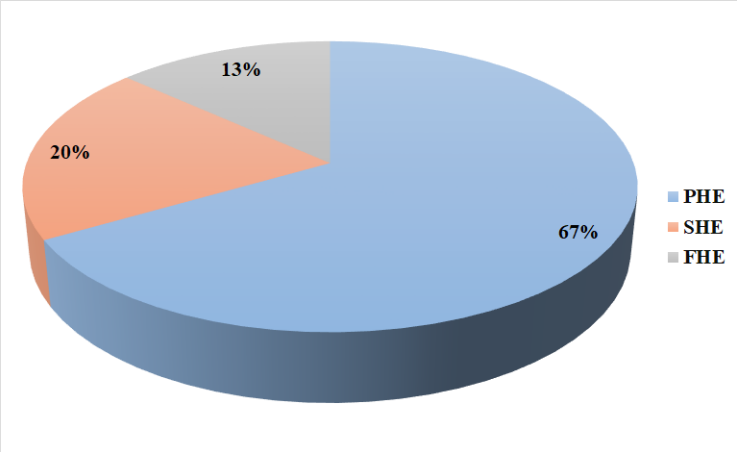


Figure 2.2: Percentage of each Homomorphic Encryption category applied to biometric recognition

The percentage distribution of machine learning classification techniques achieving PP training only, PP classification only and both PP training, PP classification discussed in the literature is shown in Figure. 2.3. We can infer from the Figure. 2.3 that 47% of works are achieved only PP training, 40% works are achieved only PP classification and 13% of works are achieved both PP training & PP classification. Therefore, machine learning classification techniques need to be applied on encrypted data in such a way that achieves both PP training & PP classification.

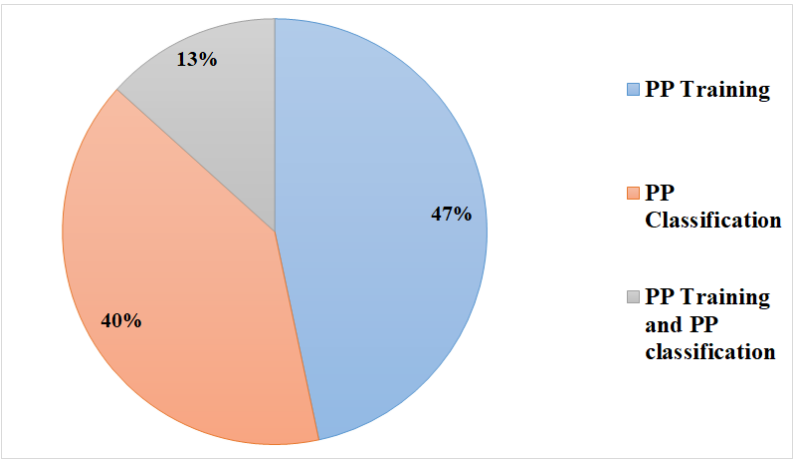


Figure 2.3: Percentage of machine learning classification techniques achieving PP training, PP classification and both discussed in the literature

2.6 Summary

In this chapter, some of the existing works on homomorphic encryption applied to biometric authentication, machine learning approaches applied to iris recognition, machine learning on encrypted data and Blockchain for biometrics are described. Most of the existing biometric authentication methods based on HE to provide privacy for the biometric templates assume that the server is "Honest-but-serious". Therefore, the existing methods only solve the modify templates attack of BAS and fail to overcome the override comparator attack of BAS. The techniques proposed in chapter 4 and chapter 5 of this thesis solve the modify templates, intercept channel and override comparator attacks of BAS. Some of the machine learning classification techniques on encrypted data provide only PP training or PP classification but not both. So, the classification techniques proposed in chapter 5 provides both PP training and PP classification.

Chapter 3

Privacy-preserving Iris Authentication on Honest-but-Curious Server (PIAHC)

The brief introduction of homomorphic encryption and the advantage of applying homomorphic encryption in biometric recognition system is explained in section 1.6.4. The main contributions of this chapter are described below:

- A privacy-preserving iris authentication system using FHE (PIAHC) is proposed to solve the limitations of biometric cryptosystems and cancelable biometrics.
- Rotation-invariant iris template is generated to solve the rotational inconsistency problem.
- An algorithm to compute the Hamming distance between the encrypted reference iris template and encrypted probe iris template is designed.

The block diagram of PIAHC is shown in Figure. 3.1. PIAHC involves two entities and three modules. The two entities are client device and server. The three modules are Generation of iris codes, Encryption/decryption and Computation of hamming distance on encrypted iris templates. The steps involved during the enrollment & authentication phases of PIAHC are described in Algorithm 3.1 and Algorithm 3.2.

Assumptions of PIAHC:

PIAHC assume the following:

- The client device has limited computation resources and memory.
- During the enrollment and authentication phases, the client device is fully trusted and stores the secret key of the user in a secure manner.
- The server is Honest-but-Curious.

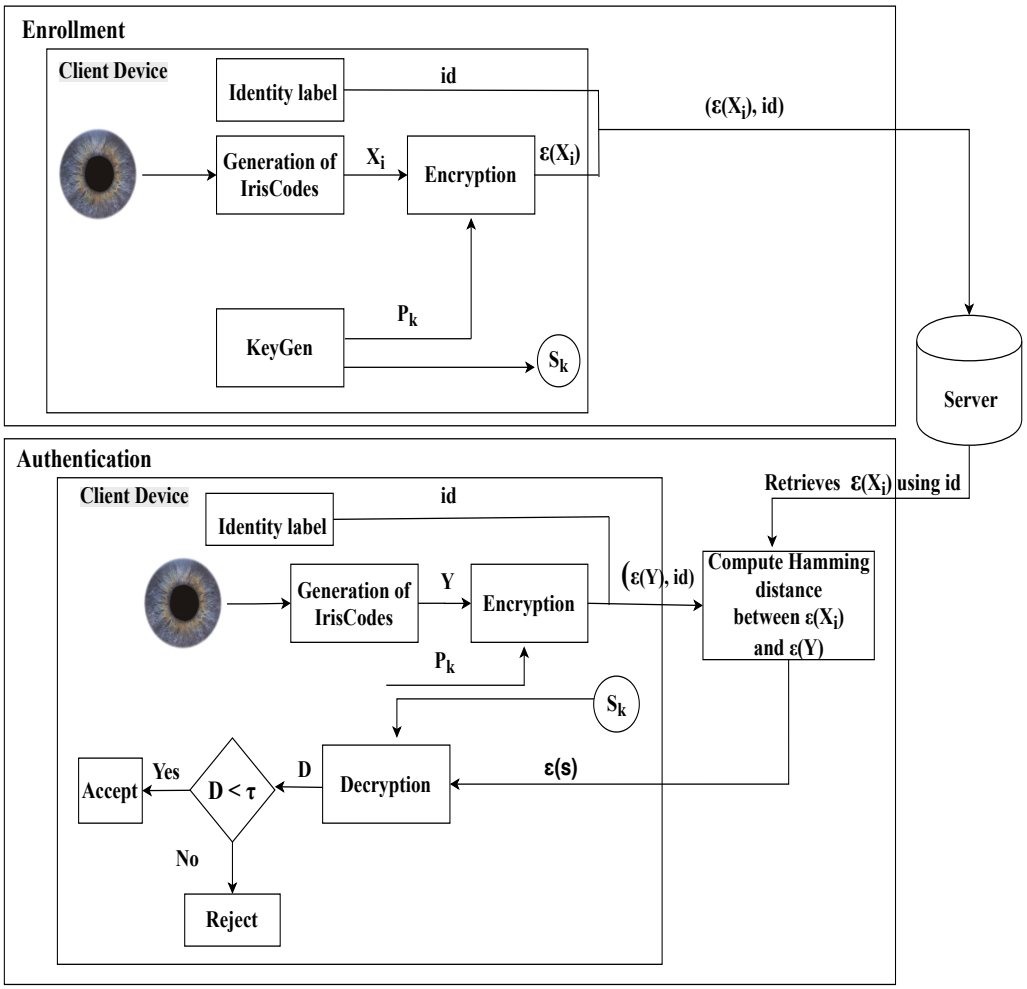


Figure 3.1: Block diagram of Privacy-preserving Iris Authentication on Honest-but-Curious Server (PIAHC)

Algorithm 3.1 Enrollment phase of PIAHC

Input: Reference iris image
Identity label, id

Output: Encrypted reference iris template, $\varepsilon(x)$

1: Client device generates the iris template from the reference iris image using University of Salzburg tool kit [58].
2: Client device generates the rotation-invariant iris code, R. It further reduces the dimensions of R as described in section 3.1.2, encode the reduced iris template as described in section 3.1.3 and obtains X_i.
3: Client device generates the secret key (S_k), public key (P_k). // *Refer section 3.2.1.1*
4: The client device encrypts the encoded iris template. //*Refer section 3.2.1.2*
5: The encrypted reference iris template, $\varepsilon(X_i)$ and identity label, id are sent to the server by the client device. The server stores ($\varepsilon(X_i)$, id) into the database.

3.1 Generation of Iris Codes

The generation of iris code comprises of three phases:

1. Generation of rotation-invariant iris template.
2. Compression of rotation-invariant iris template.
3. Encode the compressed rotation-invariant iris template using batching scheme.

The extracted iris template from the iris image has rotational inconsistency problems due to the head tilt of a person results in false accept or false reject. To overcome this limitation, rotation invariant iris template is generated in the generation of rotation-invariant iris template phase. The rotational-invariant iris template is first compressed and then encoded using the batching scheme to improve the performance of the system in terms of computational time in the compression of rotation-invariant iris template phase and encoding phase.

3.1.1 Generation of rotation-invariant iris template

The rotational irregularities caused during acquisition may affect the performance of the system. There are several techniques proposed in the spatial domain, which works on iris textures to evade the rotational inconsistency problem, and these techniques do not bring high recognition accuracy. In this scenario, PIAHC is designed to solve both rotational

Algorithm 3.2 Authentication phase of PIAHC

Input: Probe iris image
Identity label, id
Output: Accept/Reject

1: Client device generates the iris template from the probe iris image using University of Salzburg tool kit [58].
2: Client device generates the rotation-invariant iris code, Y. It further reduces the dimensions of Y as described in section 3.1.2 and encode the reduced iris template as described in section 3.1.3.
3: The client device encrypts the encoded probe iris template. *//Refer section 3.2.1.2*
4: The client device sends ($\varepsilon(Y)$, id) to the server.
5: The server retrieves the reference iris template with the same identity label from the database.
6: The server computes the hamming distance between $\varepsilon(X_i)$ & $\varepsilon(Y)$ by using Algorithm 3.3 and send result $\varepsilon(s)$ to the client device.
7: The client device decrypts $\varepsilon(s)$ by using S_k, and obtains the decrypted result, D. The client device compares D with a threshold τ, and returns accept/reject.

inconsistency problem and achieving high recognition accuracy. PIAHC shifts each of the iris template by ± 8 to get the rotation-invariant iris template. This helps in shifting in a sequence of eight columns left and right to get 16 shifted iris templates and the one original iris template. Four samples are considered per user to obtain the rotation invariant iris template. Out of four samples, one sample is considered as a reference sample. The hamming distances are calculated between the 17 iris instances taken from each sample and the considered reference iris code. The least hamming distance from each sample is considered, and the average of 3 iris instances is calculated. The template obtained by calculating the average is considered as the reference iris template.

During the verification stage, the probe iris template is shifted by ± 8. Further, the Hamming distances between the 17 probe iris instances and the reference iris instance are calculated. The iris instance having the minimum Hamming distance is considered as the probe template. The Equal Error Rate (EER) for the original iris template and the rotation-invariant iris template are shown in Table 3.1. It is observed that 4.43 is the EER obtained before applying a rotation-invariant mechanism, and 1.34 is the optimal value after the rotation-invariant mechanism is involved. It is observed from the results that a low EER is obtained with the rotation-invariant operation when compared to the template without

rotation-invariant. So, rotation-invariant iris template results in better accuracy.

Table 3.1: Comparison of EER between original iris template and rotation-invariant iris template for CASIA-V 1.0

Original iris template	**Rotation-invariant iris template**	
Equal Error Rate	**Reference Sample**	**Equal Error Rate**
4.43	1	2.28
	2	**1.34**
	3	1.86
	4	2.35

3.1.2 Compression of rotation-invariant iris template

The size of the iris template determines the performance of the system. The above phase produces an iris template of size 1 $\times$ 10240. The computational performance of the overall system can be improved by reducing the size of the iris template. So, the 10240-bit binary vector is grouped into blocks of size m by using equation (3.1). m denotes the size of the block, and we consider 4, 5, 6, 8, and 10 as m values. These m-bits are converted to decimal values and stored in a vector. The process of converting a vector of size 1 $\times$ 10240 to a vector of size 1 $\times$ 2560 is shown in Figure. 3.2. The original iris code (10240-bit) is exactly divisible with m = 4, 5, 8, and 10. But, the 10240-bit vector is grouped into 1706 blocks if the original iris code is divided with m = 6 and 2-bits will be left. Four zeros are left padded to these 2-bits, and the total 6-bits are considered as one block. Therefore, a total of 1707 blocks are obtained.

$$compressed\ iriscode\ size = \frac{Total\ number\ of\ bits}{m} \tag{3.1}$$

The EER obtained for the original 10240-bit binary vector, and different sizes of iris template are shown in Table 3.2. From Table 3.2, we can infer that the 10240-bit binary vector

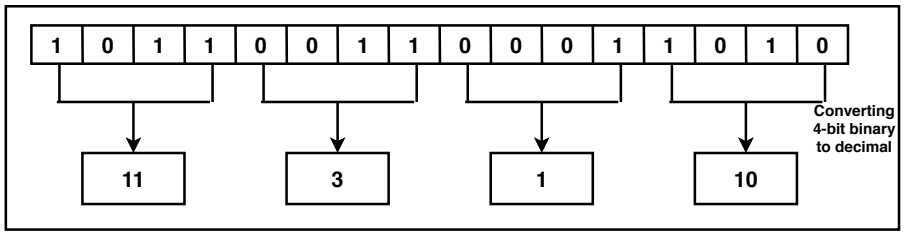


Figure 3.2: Compression of Bits($1 \times 10240 \rightarrow 1 \times 2560$)

Table 3.2: Compression of 10240 vector into blocks of various sizes for CASIA-V 1.0

Rotation-invariant iris template	**EER**	m	**compressed iris template size**	**EER**
10240	1.34	10	1024	1.23
		8	1280	0.81
		6	1707	0.79
		5	2048	0.54
		4	**2560**	**0.19**

is divided into blocks of 4 bits to achieve better performance.

3.1.3 Encode the compressed rotation-invariant iris template using Batching scheme

The input to the Brakerski/Fan-Vercauteren (BFV) HE scheme (used in section 3.2.1) is a polynomial in ring R_x, whereas the output in section 3.1.2 are integers. Encoders are accountable to convert integers into a polynomial in ring R_x. BFV scheme consists of four encoding techniques, namely scalar encoder, integer encoder, fractional encoder and Chinese Remainder Theorem (CRT) Batching [128]. CRT batching scheme performs better when compared to other encoding techniques, since it can form a single plaintext polynomial for a group of n integers modulo x. As a result, with a single instruction, an operation can be performed on multiple data simultaneously. This manner is often called as Single Instruction, Multiple Data (SIMD). Batching technique improves the performance of the system by encrypting a group of integers at once instead of encrypting a single integer

[133, 134]. Batching is based on the Chinese remainder theorem.

If Υ is the $primitive\ 2n^{th}\ root\ of\ unity\ modulo\ x$ then the polynomial modulus a^n+1 can be rewritten as

$$a^n + 1 = (a - \Upsilon)(a - \Upsilon^3)...(a - \Upsilon^{2n-1})(mod\ x)$$

If we consider plain text modulus, a to be multiplication of many small prime factors i.e., $x = \prod_{i=1}^{n} x_i$, then the ring R_x can be factorized by using the CRT as

$$R_x = \frac{\mathbb{Z}_x[a]}{a^n+1} = \frac{\mathbb{Z}_x[a]}{\prod_{i=0}^{n-1}(a - \Upsilon^{2i+1})} \approx \prod_{i=0}^{n-1} \mathbb{Z}_x$$

At the cost of single addition (multiplication) in R_x, one can perform n coefficient-wise additions (multiplications) in integers modulo x.

3.2 Ensuring the confidentiality of iris templates and Computation of Hamming Distance

3.2.1 Ensuring the confidentiality of iris templates

Basic Notations:

For $x \in \mathbb{Z}$, a ring R_x = $\mathbb{Z}_x[a]$/(a^n+1) denotes polynomials of degree smaller than n with the coefficients modulo x. $g \xleftarrow{\$} F$ represents g is sampled uniformly from the finite set F. Similarly, $g \leftarrow \chi$ represents g is sampled from a discrete truncated Gaussian. Consider the largest integer smaller than or equal to x, smallest integer greater than or equal to x and closest integer to x are denoted by $\lfloor x \rfloor$, $\lceil x \rceil$ and $\lfloor x \rceil$. The reduction of an integer by modulo x is denoted by $[.]_x$.

BFV scheme [72] is used to ensure the confidentiality of the iris templates. The security of the BFV scheme relies on the hardness of solving the Ring Learning With Errors (RLWE) problem. The main difference between symmetric or asymmetric, and HE is the evaluation function. As explained in section 1.7.2, FHE technique consists of four func-

tions, namely Key Generation (KeyGen), Encryption (Enc), Evaluation (Eval) and Decryption (Dec). The steps involved in each function are explained in the following sections:

3.2.1.1 Key Generation

The function to generate the public key, secret key and evaluation key of BFV scheme [72] is shown in Figure 3.3. The function takes the security parameter (λ) as input and produces P_k, S_k and δ_{evk} as output.

KeyGen$(1^\lambda) \Rightarrow (P_k,\ S_k,\ \delta_{evk})$

Input: Security Parameter λ

Output: Public key (P_k), Secret key (S_k) and Evaluation key (δ_{evk})

$$S_k \xleftarrow{\$} R_2$$

$$P_k = ([-(aS_k + e)]_q, a)$$

for $j = 0, 1, ..., l, where\ l = \lfloor log_\omega q \rfloor$, sample $x_j \xleftarrow{\$} R_q$, $e_j \leftarrow \chi$

$$\delta_{evk} = ([-(x_j S_k + e_j) + \omega^j S_k^2]_q\ ,\ x_j)$$

Figure 3.3: Key Generation function in BFV Homomorphic Encryption

3.2.1.2 Encryption

The function to encrypt the value in BFV scheme [72] is shown in Figure 3.4. It takes the plaintext m and public key P_k as input and produces the encrypted value of m i.e., $\varepsilon(m)$ as output.

Enc(P_k, m) $\rightarrow \varepsilon(m)$

Input: Public key (P_k), message m

Output: Encrypted message $\varepsilon(m)$ = [ct_0, ct_1]

for $m \in R_x$, let $P_k = (P_k[0]\ ,\ P_k[1])$, sample $v \xleftarrow{\$} R_2$ and $a_1, a_2 \leftarrow \chi$. The encrypted value ($\varepsilon(m)$) is given as :

$$\varepsilon(m) = ([m + P_k[0]v + a_1]_q\ ,\ [P_k[1]v + a_2]_q).$$

Figure 3.4: Encryption function in BFV Homomorphic Encryption

3.2.1.3 Evaluation (Add & Multiply)

The steps required to perform addition and multiplication of two encrypted values in BFV scheme [72] are shown in Figure 3.5. The addition of two encrypted values in BFV scheme is similar to performing the addition of two polynomials. The multiplication of two encrypted values involves two steps: the first step is multiplying two polynomials together. The limitation is that the result consists of 3 ring elements instead of 2. To solve this limitation, re-linearisation is used.

Eval(P_k, C, $\varepsilon(X)$) $\rightarrow \epsilon(R)$

Input: Public key ($P_k = (P_k[0], P_k[1])$), Evaluated Circuit C, Group of Cipher texts $\epsilon(X)$

Output: Computed result $\epsilon(R)$

Let $ct_0[0] = ([m1 + P_k[0]v + a_1]_q)$, $ct_0[1] = [P_k[1]v + a_2]_q)$, $ct_1[0] = ([m2 + P_k[0]v + a_1]_q)$, $ct_1[1] = [P_k[1]v + a_2]_q)$ and $a_1, a_2 \leftarrow \chi$.

If we want to perform addition of two encrypted values then the evaluated circuit C is add and is computed as

$$\varepsilon(R) = add(\varepsilon(m1), \varepsilon(m2)) = ([ct_0[0] + ct_1[0]]_q, [ct_0[1] + ct_1[1]]_q)$$

If we want to perform product of two encrypted values then the evaluated circuit C is $multiply$ and is computed as

$$\epsilon(R) = multiply(\epsilon(m1), \epsilon(m2)) = (p_0, p_1)$$

where $p_0 = [\lfloor \frac{x}{q}(ct_0[0]ct_1[0]) \rceil]_q + \sum_{i=1}^{l} \delta_{evk}[i][0]([\lfloor \frac{x}{q}(ct_0[1]ct_1[1]) \rceil]_q)^i$ and $p_1 = [\lfloor \frac{x}{q}(ct_0[0]ct_1[1] + ct_0[1]ct_1[0]) \rceil]_q + \sum_{i=1}^{l} \delta_{evk}[i][1]([\lfloor \frac{x}{q}(ct_0[1]ct_1[1]) \rceil]_q)^i$

Figure 3.5: Evaluation function in BFV Homomorphic Encryption

3.2.1.4 Decryption

The function to decrypt the encrypted value in BFV scheme [72] is shown in Figure 3.6. It takes the encrypted value $\varepsilon(m)$ and secret key S_k as input and produces the original value m as output.

Dec(S_k, ct) $\Rightarrow m$

Input: Secret key (S_k), Cipher text $\varepsilon(m)$

Output: message m

$$m = [\lfloor \frac{x}{q}[ct[0] + ct[1]S_k] \rceil]$$

Where $ct[0] = [m + P_k[0]v + a_1]_q$, $ct[1] = [P_k[1]v + a_2]_q$ and $a_1, a_2 \leftarrow \chi$

Figure 3.6: Decryption function in BFV Homomorphic Encryption

3.2.2 Computation of Hamming Distance on encrypted data

The additive and multiplicative properties of BFV scheme are used to compute the Hamming distance on encrypted iris templates. Given $\epsilon(x)$ and $\epsilon(y)$, one can compute $\epsilon(x + y)$ and $\epsilon(xy)$ without decryption with the help of FHE. The server is capable of performing computations, but not trustworthy; as a result with the help of FHE, we can encrypt our data and send it to the server which performs the hamming distance on the encrypted data. To compute Hamming distance, we need to perform both multiplication and addition. So, PHE and SHE fail to implement hamming distance on encrypted data.

Generally, Hamming distance is used as a distance measure to find the similarity between reference & probe templates in iris authentication system. This section describes about the computation of Hamming distance on encrypted templates as a result privacy of the iris templates is preserved. The encrypted reference & probe iris templates are denoted as $\varepsilon(a)$, $\varepsilon(b)$ and e_1 = -$\sum_{i=0}^{n-1} a^{n-i}$, e_2 = $\sum_{i=0}^{n-1} a^i$ are the constant polynomials. The Hamming distance on encrypted templates can be given as

$$Hamming\ Distance = \varepsilon(a) \times e_1 + \varepsilon(b) \times e_2 - 2 \times \varepsilon(a) \times \varepsilon(b) \tag{3.2}$$

Due to the batching scheme used in PIAHC, the Hamming distance can be computed with only four multiplications and two additions instead of 2560 homomorphic multiplications. The server sends $\varepsilon(s)$ to the client device. The client device decrypts $\varepsilon(s)$ with S_k. Client device decomposes the decrypted result using equation (3.3) and obtains the decomposed

Algorithm 3.3 Computation of Hamming distance on encrypted data

Input: $\varepsilon(a)$, $\varepsilon(b)$
Output: Encrypted Hamming distance score, $\varepsilon(d)$

begin
  $e_1 \leftarrow -\sum_{i=0}^{n-1} a^{n-i}$
  $e_2 \leftarrow \sum_{i=0}^{n-1} a^{i}$
  Encode e_1 and e_2 using Batching encoding scheme
  $ft \leftarrow multiply(\varepsilon(a),\ e_1)$ *//Batch Multiply*
  $st \leftarrow multiply(\varepsilon(b),\ e_2)$ *//Batch Multiply*
  $cst \leftarrow -2$
  Encode cst using encoding(a)=sign(x)($x_{n-1}a^{n-1} + .. + x_1x + x_0$)
  $temp \leftarrow multiply(cst, \varepsilon(a))$ *//Batch Multiply*
  $tt \leftarrow multiply(temp, \varepsilon(b))$ *//Batch Multiply*
  $res \leftarrow add(ft, st)$
  $result \leftarrow add(res, tt)$
  $\varepsilon(d) \leftarrow result$
  return $\varepsilon(d)$
end

result as $m(a) \rightarrow [m(\beta_0), m(\beta_1), .., m(\beta_{n-1})]$.

$$Decompose : R_x \rightarrow \prod_{i=0}^{n-1} \mathbb{Z}_x \tag{3.3}$$

The ratio between the number of non-zero values in the decomposed result to the total number of bits is denoted as D. The client device compares D with threshold τ to check whether the user is genuine or not.

$$Authentication = \begin{cases} Accept, & \text{if D} < \tau. \\ Reject, & \text{otherwise.} \end{cases} \tag{3.4}$$

3.3 Implementation details and Security Analysis of PI-AHC

The following measures are used to evaluate the efficiency of a biometric system according to biometric information protection [23].

1. Performance evaluation in terms of EER, d' and KS-test.

2. Irreversibility and Unlinkability Analysis.

3. Computational cost in terms of time taken to perform operations.

3.3.1 Performance Evaluation of PIAHC

Table 3.3: Comparison of EER (in terms of %) between unprotected rotation-variant, unprotected rotation-invariant and protected rotation-variant iris template

Database	Size of Iriscode	URV iris template	URI iris template	PRI iris template
CASIA-V 1.0	1024	3.83	1.23	1.23
	1280	2.98	0.81	0.81
	1707	2.30	0.79	0.79
	2048	2.28	0.54	0.54
	2560	**2.13**	**0.19**	**0.19**
CASIA-V3-Interval	1024	4.15	1.38	1.38
	1280	3.68	0.94	0.94
	1707	3.45	0.82	0.82
	2048	3.54	0.58	0.58
	2560	**3.36**	**0.39**	**0.39**
IITD	1024	4.58	2.24	2.24
	1280	4.35	2.09	2.09
	1707	4.09	1.32	1.32
	2048	4.19	1.48	1.48
	2560	**4.05**	**0.99**	**0.99**
SDUMLA-HMT	1024	3.76	1.00	1.00
	1280	3.72	0.96	0.96
	1707	3.12	0.32	0.32
	2048	2.98	0.28	0.28
	2560	**3.68**	**0.94**	**0.94**

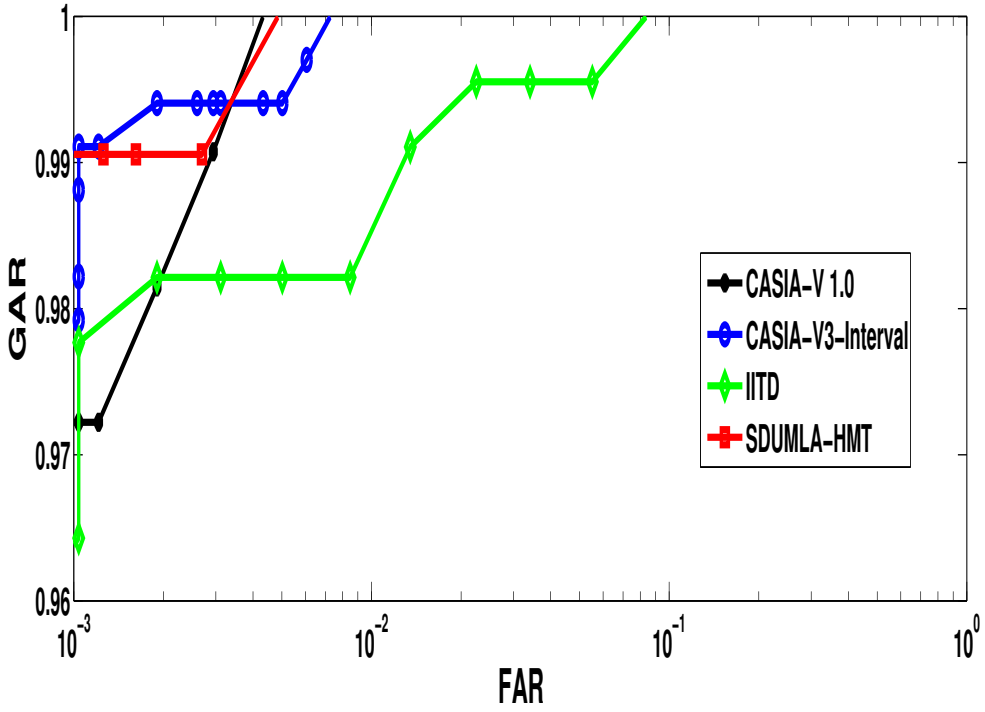
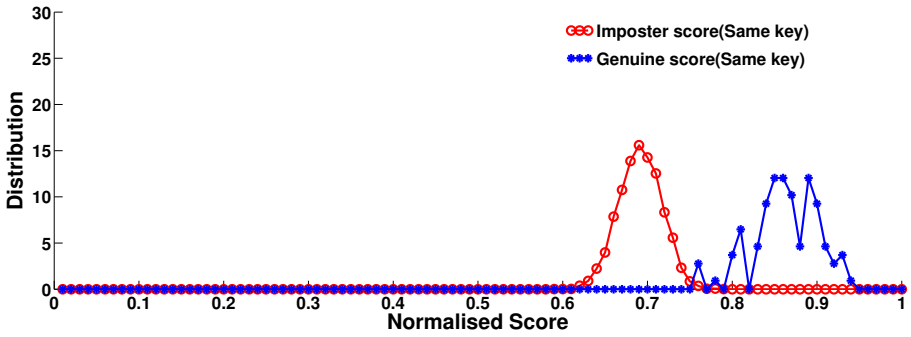


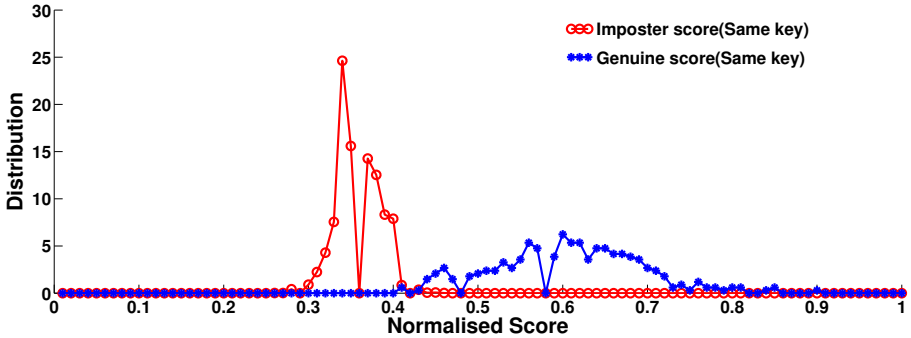
Figure 3.7: ROC Curves of PIAHC for CASIA-V 1.0, CASIA-V3-Interval, IITD and SDUMLA-HMT databases

The comparison of EER between unprotected rotation variant (URV), unprotected rotation invariant (URI), and protected rotation invariant (PRI) iris templates for different sizes are shown in Table 3.3. The protected rotation invariant iris templates indicate the templates with encryption & rotation invariant operation. The unprotected rotation invariant iris templates suggest the templates without encryption & with the rotation invariant operation, and unprotected rotation variant iris templates indicate the templates without encryption & rotation invariant operation. We can infer from Table 3.3 that there is no degradation of accuracy with PIAHC method. The increase in accuracy is due to the rotation invariant operation.

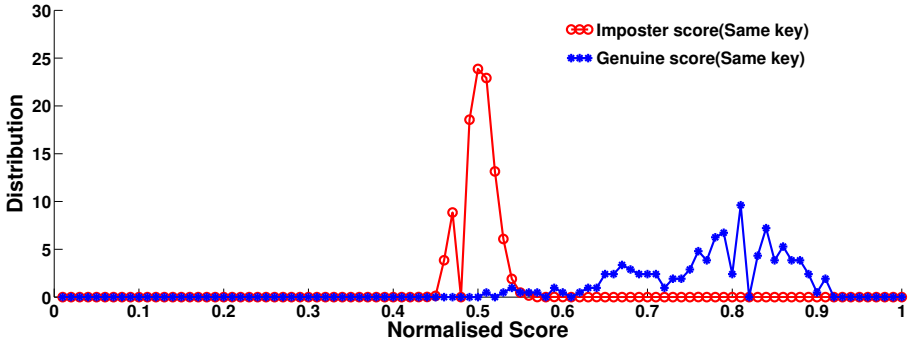
The ROC curves of PIAHC for different databases is shown in Figure 3.7. The clear separation between genuine and imposter scores for different databases are shown in Figure 3.8. The separability measures (d' & KS-test values) and EER on encrypted data for different databases are shown in Figure 3.9.



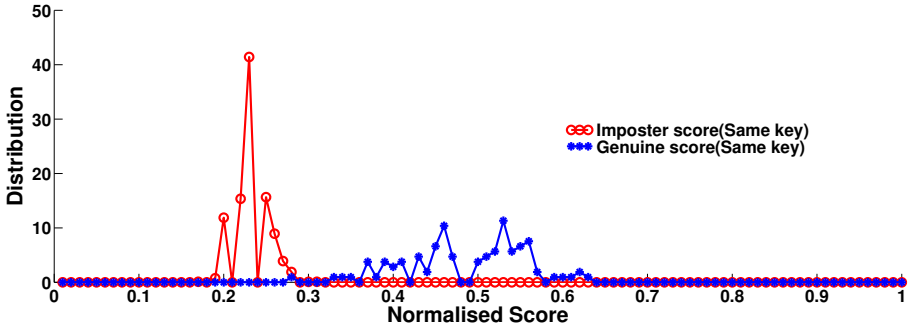
(a)



(b)



(c)



(d)

Figure 3.8: Genuine and Imposter distributions of PIAHC for (a) CASIA-V 1.0 (b) CASIA-V3-Interval (c) IITD and (d) SDUMLA-HMT databases

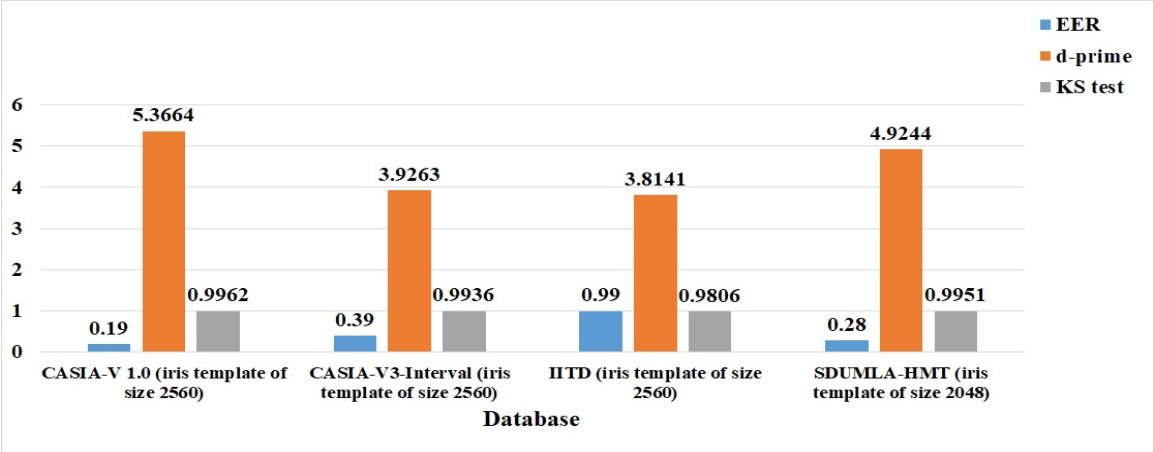


Figure 3.9: EER, Separability Measures (d′ and KS test) for CASIA-V 1.0, CASIA-V3-Interval, IITD and SDUMLA-HMT databases

3.3.2 Security Analysis of PIAHC

The template protection method must satisfy the requirements of irreversibility, revocability and unlinkability to ensure the privacy of the iris templates. The vulnerability of attacks in PIAHC can occur in the following entries:

1. The server database.
2. The client device.
3. The communication channel between the server and the client device.

The client device extracts the features of the iris image, and the secret key is also stored in the client device. Hence, security is to be ensured for the client device. As, PIAHC assume the client device is a trusted entity, the keys and features of iris image are secure. Since the security of PIAHC depends on the apparent hardness of Ring Learning with Errors (RLWE) problem, the iris templates stored in the server database are secure. It is difficult to decrypt the encrypted iris templates without the secret key. As a result, the communication channel is also reliable.

Ring Learning with Errors (RLWE) [108]: Problems like integer factorization & discrete logarithm are considered as the basis for many asymmetric cryptographic algorithms in the early 1980s. But these algorithms will have serious problems in the near future with the existence of Quantum computers [135]. RLWE is a computational problem which

serves as the support of advanced cryptographic algorithms constructed to defend against cryptanalysis by Quantum computers. The advancement to Learning with Errors (LWE) problem is referred to as RLWE and is specialized to polynomial rings over finite fields. The security of many prominent homomorphic encryption schemes relies on the LWE [136] and RLWE [108]. The RLWE problem is to solve $n(x)$ from a random noisy system.

$$\begin{cases} p_0(x)n(x) + t_0(x) = q_0(x)\ mod\ rA \\ p_1(x)n(x) + t_1(x) = q_1(x)\ mod\ rA \\ . \\ . \end{cases}$$

where $p_i(x)$ - uniformly random polynomials, $t_i(x)$ - unknown small random polynomials, r is a prime and is given as $r \equiv 1(mod\ 2n)$, n is a power of 2 and $A = \mathbb{Z}[x]/(x^n+1)$. Given $p_i(x)$ and $q_i(x)$, it is computationally infeasible to find the polynomial $n(x)$. The difficulty of solving $RLWE$ problem is similar to solving NP-Hard Shortest Vector Problem (SVP).

Irreversibility Analysis: Irreversibility refers to obtaining the original template from the encrypted template. The client device sends the encrypted reference and encrypted probe iris templates of a user to the server for distance computation. The server computes the Hamming distance on the encrypted templates and returns the encrypted result to the client device. As the PIAHC uses BFV scheme to protect the templates, and the security of BFV scheme relies on solving the RLWE problem, it is computationally infeasible to decrypt the templates by the server or an imposter without secret key (S_k). Therefore, PIAHC satisfies the irreversibility property.

Revocability Analysis: Revocability ensures that a new protected template should be generated by the protection method if the old template is compromised or stolen. In PIAHC, Revocability can be achieved by re-encrypting the samples in the database with a new key pair (P'_k, S'_k) instead of acquiring the new samples from the users.

Unlinkability Analysis: Unlinkability ensures that there won't be any correlation between

the protected templates used in different applications. BFV scheme used in PIAHC is based on probabilistic encryption. Due to the randomness involved in BFV scheme, different ciphertexts can be generated even if the same message is encrypted multiple times with the same key, and there won't exist any similarity between the generated ciphertexts.

Table 3.4: Total time taken in PIAHC (with & without batching scheme)

λ[1]	M[2]	NFT[3]	Parameters			Batching Time(Seconds)				NBT(Seconds)[4]			
			n	q (bits)	x	Enc	Score	Dec	Total	Enc	Score	Dec	Total
128-bit	640	0.0075	1024	29	40961	0.0014	0.0026	0.0008	0.0048	0.7	1.5	0.1	2.3
	1024	0.014	2048	56	40961	0.0026	0.0052	0.0016	0.0094	2.1	4.8	0.5	7.4
	1280	0.019	2048	56	40961	0.0027	0.0053	0.0016	0.0096	2.6	6.0	0.6	9.2
	1707	0.021	2048	56	40961	0.0028	0.0053	0.0018	0.0099	3.5	8.0	0.8	12.3
	2048	0.035	4096	110	40961	0.0060	0.018	0.0039	0.0279	10.0	35.2	3.2	48.4
	2560	0.044	4096	110	40961	0.0058	0.019	0.0038	0.0286	12.5	44.0	4.1	60.6
192-bit	640	0.0075	1024	20	40961	0.0013	0.0027	0.0009	0.0049	0.7	1.4	0.1	2.2
	1024	0.014	2048	39	40961	0.0026	0.0053	0.0015	0.0094	2.3	5.2	0.5	8.0
	1280	0.019	2048	39	40961	0.0027	0.0055	0.0016	0.0098	2.8	6.5	0.6	9.9
	1707	0.021	2048	39	40961	0.0029	0.0053	0.0016	0.0098	3.8	8.7	0.9	13.4
	2048	0.035	4096	77	40961	0.0063	0.018	0.0041	0.0284	10.9	34.8	3.5	49.2
	2560	0.044	4096	77	40961	0.0059	0.019	0.0040	0.0289	13.6	43.5	4.4	61.5

[1] λ refers to security.
[2] M refers to size of iris template.
[3] NFT refers to Time in seconds without FHE.
[4] NBT refers to Time in seconds without Batching.
Enc, Score and Dec stands for time taken to perform encryption, hamming distance between probe and reference templates, and decryption.

3.3.3 Computational Analysis of PIAHC

The security parameters used in PIAHC are polynomial modulus ($a^n + 1$), coefficient modulus (q), plaintext modulus (x) and security level (λ). PIAHC considered two different values for λ. From Table. 3.4, it can be inferred that the higher security level has nearly no influence on the execution time. $a^n + 1$ must be a power-of-2 cyclomatic polynomial. The security level is directly proportional to the polynomial modulus. On the other hand,

Table 3.5: Comparison Analysis in terms of Time

Method	**Iris Bit Size**	**Total time (in secs)**	**Homomorphic Encryption Scheme**
Barni, M, *et al.* [76]	6400 Bits	0.12	Damgard-Jurik cryptosystem (SHE)
Alberto Torres, W. A *et al.* [137]	2048 Bits	645.049	Lattice-based-FHE
Cheon, Jung Hee, *et al.* [138]	2400 Bits	0.57	BGV Scheme (SHE)
Kulkarni, Rohan *et al.* [139]	2048 Bits	58	BGN Cryptosystem (SHE)
Yasuda, Masaya, *et al.* [66]	2048 Bits	0.01243 (HD=0.05)	Polynomial-LWE (SHE)
PIAHC	**2560 Bits**	**0.0286 (HD=0.019)**	**BFV Scheme (FHE)**

larger $a^n + 1$ makes ciphertext size larger, and all operations become slower. n value must be a power of 2 and greater than the size of the iris template. So, PIAHC choose different n values for different sizes of iris templates. The default q values for different n values are mentioned in [128]. x can be any positive integer, and mostly it is a power of two. But, batching encoding only works when a is chosen to be a prime number and congruent to $1 \ (mod \ 2n)$. So, PIAHC considered plaintext modulus as 40961.

For a given desired security level (λ), Table. 3.4 illustrates the time taken (in seconds) to encrypt, decrypt and to compute the Hamming distance on the encrypted data for different security parameter values (n, x and q) and iris code sizes. PIAHC considered the average time in seconds by running the experiments ten times. The table also shows the time taken to compute the Hamming distance on original values. The iris template size is proportional to the computational time. PIAHC converts 1×10240 into 1×640, 1×1024, 1×1280, 1×1707, 1×2048 and 1×2560 respectively. Even though the total time taken for iris code of size 640, 1024, 1280, 1707 and 2048 is less when compared to iris code of size 2560, but the optimal accuracy is achieved with iris template of size 1×2560.

Table 3.6: Comparison of PIAHC with existing approaches (EER in terms of %)

Database		EER
CASIA-V 1.0	Dwivedi, R. *et al.* [140]	0.37
	Punithavathi, P *et al.* [141]	1.2
	Mahesh, M. *et al.* [142]	0.57
	Gad, R. *et al.* [143]	0.299
	Barni, M. *et al.* [76]	2.08
	PIAHC	**0.19**
CASIA-V3-Interval	Dwivedi, R. *et al.* [140]	0.43
	Lai, Y.L. *et al.* [144]	0.54
	Punithavathi, P *et al.* [141]	1.9
	Soliman, R.F *et al.* [145]	0.63
	Zhao, D. *et al.* [146]	1.03
	Barpanda, S.S *et al.* [147]	11.75
	Sadhya, D. *et al.* [148]	0.105
	Soliman, R.F *et al.* [149]	0.36
	PIAHC	**0.39**
IITD	Punithavathi, P *et al.* [141]	3.3
	Barpanda, S.S *et al.* [147]	12.69
	Gomez-Barrero, M. *et al.* [150]	0.7
	Sadhya, D. *et al.* [148]	1.4
	PIAHC	**0.88**
SDUMLA-HMT	Gad, R. *et al.* [143]	0.300
	Kamalskar, C *et al.* [151]	2.5947
	PIAHC	**0.28**

Table 3.7: Comparison of PIAHC with other approaches (in terms of Separability measure (d'))

	CASIA-V 1.0	CASIA-V3-Interval	IITD
Barpanda, S.S *et al.* [147]	-	1.71	1.76
Sadhya, D. *et al.* [148]		2.39	2.92
Walia, G.S. *et al.* [152]	2.6053	-	1.9578
PIAHC	**5.3664**	**3.9263**	**3.8141**

3.3.4 Comparison Analysis

PIAHC is compared with other state-of-the-works (in terms of computational time) and is given in Table 3.5. The batching scheme used in PIAHC makes the system to give a fair performance when compared to other works. On the other hand, PIAHC uses a higher bit security level, i.e., 128-bit & 192-bit, whereas the other methods use 80-bit security. From Table 3.5, we can observe the performance of Yasuda *et al.* is better when compared to PIAHC. The reason for the degradation of performance is that they used SHE whereas PIAHC uses FHE.

The EER comparison of PIAHC with state-of-the-art works is shown in Table 3.6. We can infer that PIAHC shows better EER value when compared to other existing works. The comparison of d' with the existing approaches is shown in Table 3.7. We can infer from Table 3.7 that the genuine and imposter scores are well separated when compared to other works.

3.4 Summary

In this chapter, a privacy-preserving iris authentication system using FHE (PIAHC) is proposed to preserve the privacy of iris templates by performing the matching on the encrypted iris templates. PIAHC solves the rotational inconsistency problems occurred due to the head tilt of a person by generating the rotation-invariant iris templates. These templates help to improve recognition accuracy. The rotation-invariant iris template is first compressed and then encoded using the batching scheme to improve the performance of the system in terms of the computational time. A procedure to compute the Hamming distance is proposed, which helps to check whether the user is genuine or not. PIAHC consumes 0.0185 seconds only with no performance degradation. Experimental results prove the significance and validity of PIAHC.

Chapter 4

Privacy-preserving Multi-Instance Iris Authentication on Untrusted Cloud Server using PHE schemes

A brief introduction, advantages and types of multi-biometric systems are explained in section 1.8. In particular, the multi-instance systems have many benefits like cost-effective and do not require the additional sensors, need of matching algorithms, and feature extraction methods. On the other hand, the literature reveals that the privacy-preserving schemes based on HE assume that the server/cloud server is Honest-but-curious. However, due to financial or timing reasons, the server/cloud server assigned to a task may not honestly perform the computation. The cloud server may return an arbitrary result which leads to false accept or false reject. The main contributions of this chapter are described below:

- Proposed a Blockchain-based multi-instance iris authentication system (BMIAE), which integrates ElGamal HE [39] with Blockchain technology to achieve privacy of iris templates and trust on the comparator result. The challenges of using Blockchain in biometrics are also addressed in BMIAE.
- Proposed a secure and verifiable multi-instance iris authentication using public auditor (SviaPA), which not only provides privacy for the iris templates but also includes a verification procedure to check whether the comparator result is correct or not.

- A method for secure and verifiable multi-instance iris authentication using Blockchain (SviaB) is proposed. SviaB combines Blockchain technology with Paillier HE [42]. Paillier HE provides confidentiality for the iris templates. The Blockchain provides the integrity of the encrypted reference iris templates as well as the trust of the comparator result. In addition, SviaB reduces the time taken to authenticate a person when compared to BMIAE.

4.1 BMIAE: Blockchain-based Multi-Instance Iris Authentication using Additive ElGamal Homomorphic Encryption

The flow diagram for BMIAE is shown in Figure. 4.1. BMIAE involves three entities, namely client device, server and a Blockchain network. The steps involved during the enrollment & authentication phases of BMIAE are described in Algorithm 4.1 and Algorithm 4.2.

Assumptions of BMIAE:

BMIAE assume the following:

- During the enrollment/authentication phase, the client device is fully trusted and stores the user's secret key securely at its local storage.
- The client device has limited memory and computational resources.
- The server & client device need not store the entire ledger of the Blockchain network.
- The consensus algorithm of the Blockchain is secure & robust against security attacks of the Blockchain.
- The contract address of the smart contract is shared with the server & the client device before the enrollment phase.

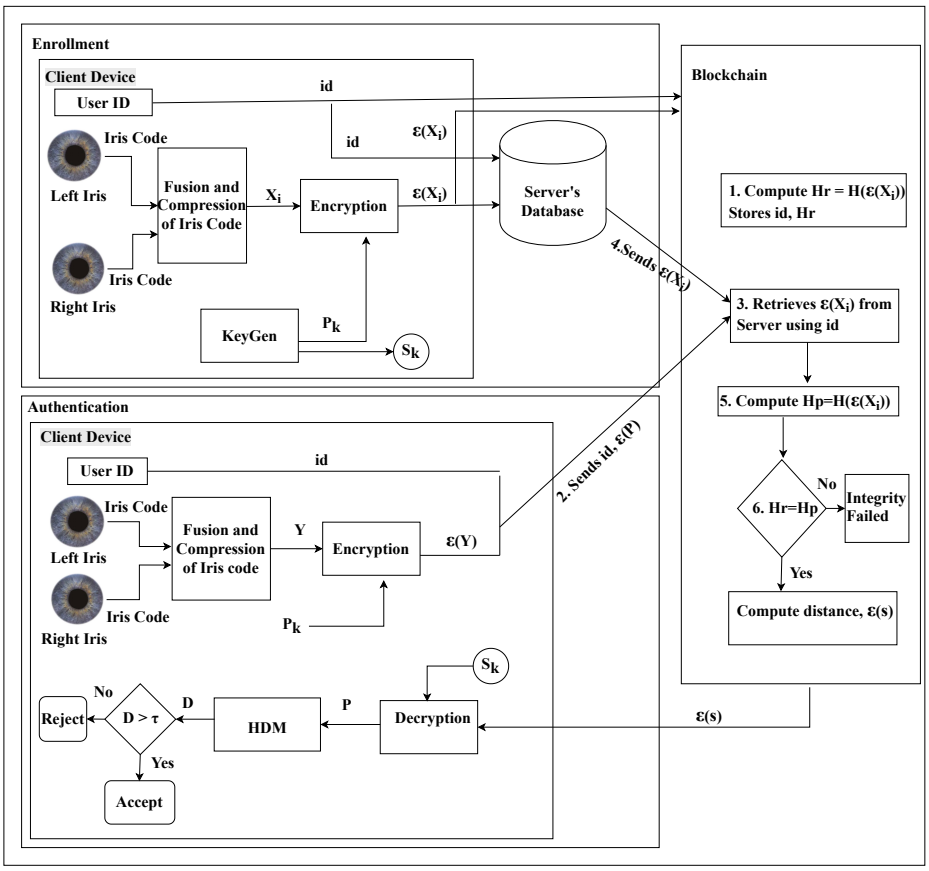


Figure 4.1: BlocK diagram of Blockchain-based Multi-Instance Iris Authentication using Additive ElGamal Homomorphic Encryption (BMIAE).

4.1.1 Generation of Integer vector from iris templates

This section comprises of three phases:

1. Fusion of left and right iris template
2. Compression of fused iris template.
3. Mapping of compressed iris template to integer vector.

The left & right iris templates extracted from the iris images using University of Salzburg tool kit [58] are fused in the first phase to achieve better recognition accuracy. The fused iris template is compressed in the second phase to improve the performance of the system in terms of computational time. In general, a number to be encrypted using ElGamal HE must present in the group $\mathbb{Z}_Q^*$, Q is a prime number. Therefore, the compressed fused template is mapped to an integer vector in the third phase.

Algorithm 4.1 Enrollment phase of BMIAE

Input: Reference images of both left & right eye
Identity label, id

1: Client device generates the iris templates from the reference left & right iris images using University of Salzburg tool kit [58].
2: Client device generates the reference fused iris template, X_i as described in section 4.1.1.1. It further reduces the dimensions of X_i as defined in section 4.1.1.2 and map the fused compressed iris template to an integer vector as described in section 4.1.1.3.
3: Client device generates the secret key (S_k) and public key (P_k). *//Refer section 4.1.2.1*
4: Client device encrypts the mapped iris template using P_k and generates the encrypted reference fused compressed iris template, $\varepsilon(X_i)$. *//Refer section 4.1.2.2*
5: The client device send ($\varepsilon(X_i)$, id) to the Blockchain & server.
6: The Blockchain calculates the hash value of $\varepsilon(X_i)$, Hr and stores (Hr, id). (BMIAE stores only Hr in Blockchain & $\varepsilon(X_i)$ in the server to overcome the storage limitations of Blockchain).

4.1.1.1 Fusion of left and right iris template

The dimension of the extracted left (L_i) and right (R_i) iris templates is 1×10240. The concatenation of L_i and R_i as shown in equation (4.1) gives the fused iris template.

$$Z = \begin{pmatrix} L_i \\ \\ R_i \end{pmatrix} \tag{4.1}$$

The dimension of the fused iris template, Z is 1×20480.

4.1.1.2 Compression of fused iris template

The size of the iris template determines the performance of the system. The fusion phase produces an iris template of size 1×20480. The computational performance of the overall system can be improved by reducing the size of the iris template. So, the 20480-bit binary vector is grouped into blocks of size v by using equation (4.2). v denotes the size of the block, and BMIAE consider 2, 4, 8, 16 and 32 as v values. The v bits obtained in each block is converted to integers and perform modulo operation on each integer by 2. The resultant binary vector is considered as the compressed fused iris template. The compression process

Algorithm 4.2 Authentication phase of BMIAE

Input: Probe images of both left & right eye
Identity label, id
Output: Accept/Reject

1: Client device generates the iris templates from the probe left and right iris images using University of Salzburg tool kit [58].
2: Client device generates the probe fused iris template, Y. It further reduces the dimensions of Y as described in section 4.1.1.2 and map the fused compressed iris template as described in section 4.1.1.3.
3: The client device encrypts the mapped fused probe iris template and generates the encrypted probe fused compressed iris template, $\varepsilon(Y)$. *//Refer section 4.1.2.2*
4: The client device sends ($\varepsilon(Y)$, id) to the Blockchain.
5: The Blockchain retrieves $\varepsilon(X_i)$ with the same identity label from the server.
6: The hash value of retrieved $\varepsilon(X_i)$, Hp is computed by the Blockchain and compares Hr, Hp. The Blockchain computes the distance $\varepsilon(s)$ if the hash values are same otherwise it will send an "integrity failed" message to the client device.
7: The client device decrypts $\varepsilon(s)$ by using S_k, and obtains the decrypted result, Y. The client device computes the number of zeros in Y and obtains D. The client device compares D with a threshold τ, and returns accept/reject.

is shown in Figure. 4.2.

$$Size\ of\ compressed\ iris\ template = \frac{Total\ number\ of\ bits}{v} \tag{4.2}$$

The EER obtained for the original 20480-bit binary vector, and different sizes of iris template are shown in Figure. 4.3. From Figure. 4.3, we can infer that the 20480-bit binary vector is divided into blocks of 16-bits for CASIA-V3-Interval, 8-bits for IITD and 8-bits for SDUMLA-HMT iris databases to achieve better performance.

4.1.1.3 Mapping of compressed iris template to integer vector

The number to be encrypted using ElGamal HE must present in the group $\mathbb{Z}_Q$, Q is a prime number. So, the compressed fused template obtained in the section 4.1.1.2 is mapped to integers which belong to $\mathbb{Z}_Q$. On the other hand, an additive HE scheme must be used to compute the distance. Generally, ElGamal is a multiplicative HE scheme but if we consider g^m instead of m, where m and g denotes the number to be encrypted and generator of the group then it satisfies the additive property. The steps involved in the conversion of binary

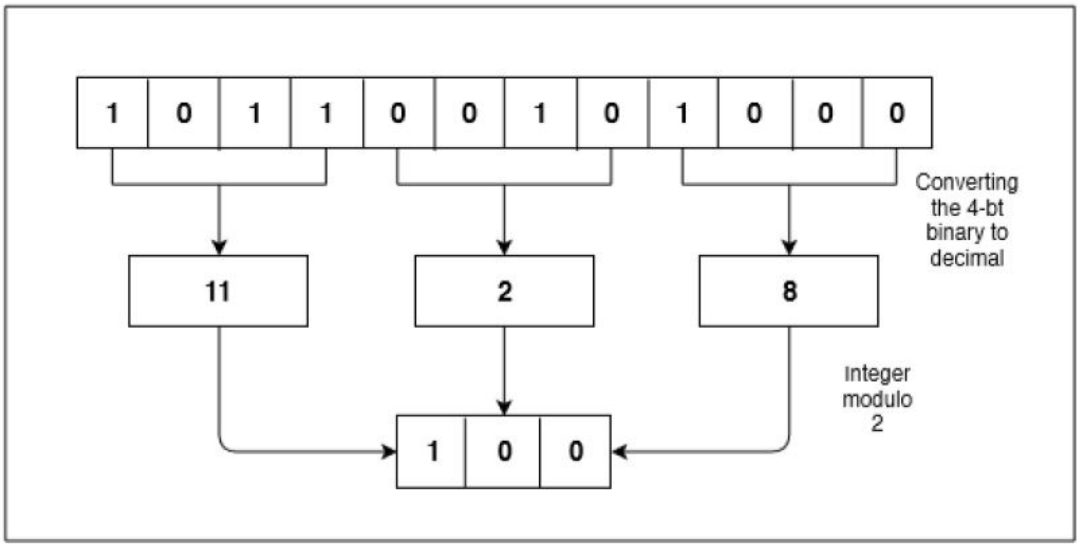


Figure 4.2: Compression of Bits in Blockchain-based Multi-Instance Iris Authentication using Additive ElGamal Homomorphic Encryption

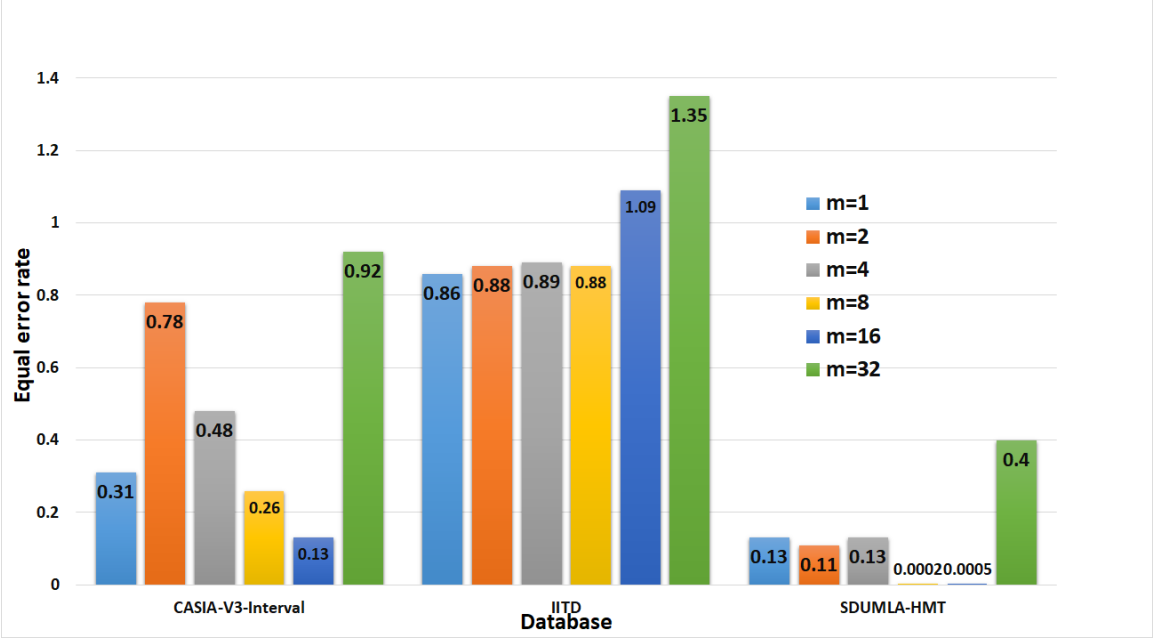


Figure 4.3: Comparison of Equal Error Rate for various sizes of iris template

to integer vector are mentioned below:

Step 1: Select a large prime Q.

Step 2: Let g be the generator of the group $\mathbb{Z}_Q$.

step 3: Two different prime numbers a_1 & a_2 such that a_1 & $a_2 < Q$, $a_1 > a_2$ are chosen randomly. The ones and zeros in the binary vector obtained in section 4.1.1.2 are replaced with g^{a_1} & g^{a_2}.

The mapping process is explained with an example. Consider Q = 131. g = 2 is the generator of the group $\mathbb{Z}_{131}$. Choose a_1 = 5 & a_2 = 2 are the two primes. Then, the ones and zeros in the binary vector are replaced with $g^{a_1} = 2^5 = 32$ and $g^{a_2} = 2^2 = 4$.

4.1.2 Ensuring the Confidentiality of Iris templates using ElGamal Homomorphic Encryption

During the enrollment and authentication phase, ElGamal HE [39] is used to ensure the confidentiality of the iris templates. The security of the ElGamal HE depends on the hardness of solving the discrete logarithm problem on a cyclic group. Generally, ElGamal HE satisfies the multiplicative property. An additive HE scheme must be used to compute the distance. In the literature, there exist various PHE schemes which satisfy the additive property. Due to its advantages over other HE schemes [39, 153], BMIAE uses a modified version (i.e., consider the message (m) to be encrypted as g^m instead of m) of ElGamal HE which satisfies the additive property. The template obtained in section 4.1.1.3 is encrypted using the Enc function given in Figure. 4.5.

ElGamal HE scheme consists of four functions, namely Key Generation (KeyGen), Encryption (Enc), Evaluation (Eval) and Decryption (Dec). The steps involved in each function are explained in the following sections:

4.1.2.1 Key Generation

The function to generate the public key, secret key of ElGamal scheme [39] is shown in Figure. 4.4. The function takes a prime number (Q) as input and produces P_k, S_k as output.

(P_k, S_k) ⟵ KeyGen(Q)

Input: Prime number Q

Output: Secret key (S_k) and Public key (P_k)

- Let g be the generator of the cyclic group $\mathbb{Z}_Q^*$ and f be the number of elements in $\mathbb{Z}_Q^*$.
- An element $r \in \{1, 2, ..., f\text{-}1\}$ is chosen randomly.
- Calculate $q = g^r \pmod{Q}$

$$(S_k) : [r].$$

$$(P_k) : [\mathbb{Z}_Q^*,\ r,\ g, q].$$

Figure 4.4: Key Generation function in ElGamal Homomorphic Encryption

4.1.2.2 Encryption

The function to encrypt the value in ElGamal scheme [39] is shown in Figure. 4.5. It takes the plaintext m and public key P_k as input and produces the encrypted value of m i.e., $\varepsilon(m)$ as output.

$\varepsilon(m) \longleftarrow$ **Enc**(P_k, m)

Input: Public key (P_k), message m

Output: Encrypted message $\varepsilon(m)$ = [ct_0, ct_1]

- An element $a \in \{1, 2, ..., f\text{-}1\}$ is chosen randomly.
- Calculate $ct_0 = g^a \pmod p$
- $s = q^a$ is the shared secret key.
- Compute $ct_1 = g^m \times s \pmod Q$

$$\varepsilon(m) = [ct_0,\ ct_1] = [g^a\ (mod\ Q), g^m \times s\ (mod\ Q)]$$

Figure 4.5: Encryption function in ElGamal Homomorphic Encryption

4.1.2.3 Evaluation (Add)

The steps required to perform addition of two original values in ElGamal scheme [39] is shown in Figure. 4.6. The addition of two original values can be obtained by the decryption of multiplication of two encrypted values.

$\varepsilon(R) \longleftarrow$ **Eval**($\varepsilon(a)$, $\varepsilon(b)$, Q)

Input: Prime number Q, $\varepsilon(a) = (g^{a_1},\ g^a \cdot q^{a_1})$, $\varepsilon(b) = (g^{a_2},\ g^b \cdot q^{a_2})$, where a_1, a_2 are the random values chosen during encryption.

Output: Computed result $\varepsilon(C)$

The addition of two values can be computed as

$$\epsilon(C) = add(a, b) = \varepsilon(a) \cdot \varepsilon(b)\ mod\ Q$$

Figure 4.6: Evaluation function in ElGamal Homomorphic Encryption

4.1.2.4 Decryption

The function to decrypt the encrypted value in ElGamal scheme [39] is shown in Figure. 4.7. It takes the encrypted value $\varepsilon(m)$ and secret key S_k as input and produces the original value m as output.

$m \longleftarrow$ **Dec(**S_k**,** $\varepsilon(m)$**)**
Input: Secret key (S_k), Encrypted message $\varepsilon(m)$ **Output**: message m' $$m' = (ct_1 \times (ct_0)^{Q-1-r})\ mod\ Q)\ mod\ Q$$ Since, we considered the message as g^m instead of m to achieve the additive property. So, discrete logarithm has to be applied on m' to obtain m.

Figure 4.7: Decryption function in ElGamal Homomorphic Encryption

Additive Property of ElGamal:

Let $\varepsilon(a)$ = $Enc(P_k, a)$ & $\varepsilon(b)$ = $Enc(P_k, b)$ be the encrypted values for messages g^a and g^b. As defined in section 1.7.1, the additive property states that the addition of two original values can be obtained by the decryption of multiplication of two encrypted values and is given in equation (4.3).

$$D_{S_k}(\varepsilon(a) \cdot \varepsilon(b)) = a + b \tag{4.3}$$

proof:

$$\begin{aligned}
\varepsilon(a) \cdot \varepsilon(b) &= (g^{a_1},\ g^a \cdot q^{a_1}) \cdot (g^{a_2},\ g^b \cdot q^{a_2}) \\
&= (g^{a_1+a_2},\ g^a \cdot g^b \cdot q^{a_1+a_2}) \\
&= (g^{a_1+a_2},\ g^{a+b} \cdot q^{a_1+a_2}) \\
&= \varepsilon(a+b)
\end{aligned}$$

4.1.3 Ensuring the integrity of encrypted reference templates and trust on computed distance using Smart contract

The smart contract running on a Blockchain network helps the BMIAE to address the override comparator attack of BRS (Refer Figure. 1.3). Therefore, the user or client device can ensure the computed distance is correct without including any third party or centralized server. The integrity of the encrypted reference iris templates is also ensured in BMIAE by comparing the hash values in Blockchain. The formal smart contract to ensure the trust on the computed distance & integrity of encrypted reference iris template is given in Figure. 4.8.

Contract-EIRTDC

Init: Set $refer := [\,]$, $Buff := Null$, $\delta_c = 0$

Enrollment: *//Computation of Hash value*

Upon receiving ("reference", $\varepsilon(X_i)$, id) from Client device

set $refer[id] := H(\varepsilon(X_i))$

Verification: Upon receiving ("verify", $\varepsilon(Y)$, id) from Client device

set $Buff := \varepsilon(Y)$

set $\delta_c := \delta + D\ (Threshold\ time)$

send (retrieve $\varepsilon(X_i)$", id) to server.

Computation: Upon receiving ("computation", $\varepsilon(X_i)$, id) from Server

require $\delta < \delta_c$

if $H(\varepsilon(X_i)) == refer[id]$

//Distance computation

set $\varepsilon(s) := \varepsilon(X_i) * Buff$

send $\varepsilon(s)$ to client device

else

send("Integrity failed") to client device

Timer: if $\delta > \delta_c$

Send("Session Expired") to Client device

Figure 4.8: Contract-Ensuring the Integrity of Reference templates and Trust on Distance Computation (EIRTDC)

4.1.3.1 Ensuring the integrity of encrypted reference iris templates

The client device sends ($\varepsilon(X_i)$, id) to server & smart contract during the enrollment phase and invokes the $Enrollment$ function of a smart contract. The server stores ($\varepsilon(X_i)$, id). The hash value of $\varepsilon(X_i)$, $Hr = H(\varepsilon(X_i))$ is computed and stores Hr in $refer[id]$ by the smart contract. The limitations of using Blockchain in biometrics like expensive storage cost and privacy are described in [119]. To solve expensive storage cost limitation, BMIAE stores only hash value of encrypted reference iris template instead of $\varepsilon(X_i)$. To overcome the privacy limitation, BMIAE encrypts the fused templates using ElGamal HE before sending the template to Blockchain.

The client device sends ($\varepsilon(Y)$, id) to a smart contract and invokes the $Verification$ function during the authentication phase. The smart contract requests the server for $\varepsilon(R)$ with the same identity label id. If the server sends $\varepsilon(Y)$ within a stipulated time, δ_c then the $Computation$ function of a smart contract is invoked otherwise $Timer$ function of smart contract gets executed. It sends "Session Expired" message to the client device. When the computation function is invoked, the smart contract computes hash value $Hp = H(\varepsilon(Y))$. The smart contract computes the distance between $\varepsilon(X_i)$ & $\varepsilon(Y)$, if the values of Hr & Hp are same otherwise it indicates that $\varepsilon(X_i)$ is modified by the intruder. Therefore, a smart contract helps to check the integrity of the encrypted reference template.

4.1.3.2 Encrypted distance computation in the Blockchain

The smart contract computes the distance; as a result, the trust on the distance is achieved. The smart contract computes the distance only if Hr and Hp are same. The distance between X_i and Y can be computed by using equation (4.4).

$$D_{man} = |X_i - Y| \tag{4.4}$$

The distance between encrypted reference and probe iris templates can be calculated by using equation 4.3.

$$
\begin{aligned}
D_{man} &= |X_i - Y| \\
&= |X_i + (-Y)| \\
\varepsilon(D_{man}) &= \varepsilon(X_i) \cdot \varepsilon(-Y)
\end{aligned}
$$

$$\varepsilon(D_{man}) = \varepsilon(X_i) \cdot \varepsilon(-Y) \tag{4.5}$$

The smart contract computes the distance between $\varepsilon(X_i)$ & $\varepsilon(Y)$ by using equation (4.5).

$\varepsilon(s) = (\varepsilon(X_i[1])\cdot\varepsilon(Y[1]),\ \varepsilon(X_i[2])\cdot\varepsilon(Y[2]),\ ...,\ \varepsilon(X_i[M])\cdot\varepsilon(Y[M])) = (\varepsilon(s_1), \varepsilon(s_2), ..., \varepsilon(s_M))$

The smart contract send $\varepsilon(s)$ to the client device.

4.1.4 Computation of Hamming distance (from $\varepsilon(s)$) (HDM)

Step 1: Client device decrypts $\varepsilon(s)$ by using S_k and obtains P = (s_1, s_2, . . ., s_M).

Step 2: Compute a vector, R consisting of remainder values obtained by performing modulus operation on P with $g^{a_1} \times g^{a_2}$. R_i = P_i mod a where i = 1, 2, 3,..., M and a = $(g^{a_1}) \times (g^{a_2})$

R = $[r_1, r_2, r_3, ..., r_M]$.

Step 3: A binary vector H is computed by using equation (4.6).

$$H_i = \begin{cases} 0, & \text{if } r_i\text{=0.} \\ 1, & \text{otherwise.} \end{cases} \tag{4.6}$$

Step 4: Client device computes the number of zeros in the vector, H. Number of zeros helps in calculation of Hamming distance between the reference and probe templates.

$$Hamming\ distance\ (D) = 1 - \frac{Number\ of\ zeros\ in\ H}{Total\ number\ of\ bits\ in\ H} \tag{4.7}$$

Step 5: The result D is compared with τ to decide whether the user is genuine or not.

$$Authentication = \begin{cases} Accept, & \text{if D} > \tau. \\ Reject, & \text{otherwise.} \end{cases} \tag{4.8}$$

4.1.5 Limitations of BMIAE

In BMIAE, the Blockchain computes the distance between the encrypted reference iris template and encrypted probe iris template to ensure the trust on the computed result. The limitations of BMIAE are as follows:

- The computational cost and execution time required to authenticate a person is more.
- The size of the iris template varies from each database in BMIAE to obtain optimal accuracy. Therefore, BMIAE needs to find the size of the iris template for every database, which is a cumbersome process.

4.2 SviaPA: Secure and Verifiable Multi-Instance Iris Authentication using Public Auditor

SviaPA is the first known multi-instance iris authentication system which provides privacy to the user data, i.e., iris templates as well as trust on the computed result. The flow diagram of the SviaPA is shown in Figure. 4.9. SviaPA consists of four entities, namely client device, trusted authenticator, cloud server and public auditor. The role of trusted authenticator is to 1) Generate the public (P_k) and secret (S_k) keys used in the encryption. 2) Reduce the dimensions of fused iris template by using autoencoders and 3) Send accept/reject decision to the client device. The cloud server provides the storage and computation resources to the client device. If the cloud server is malicious, then the imposter may get access to the system, which is a severe problem. So, we introduce a public auditor as a third party who helps to check the correctness of the result returned by the cloud server and send the verification result to the trusted authenticator. The trusted authenticator determines whether the

user is genuine or not based on the verification result. The steps involved in the enrollment and authentication phases are shown in Algorithm 4.3 and Algorithm 4.4.

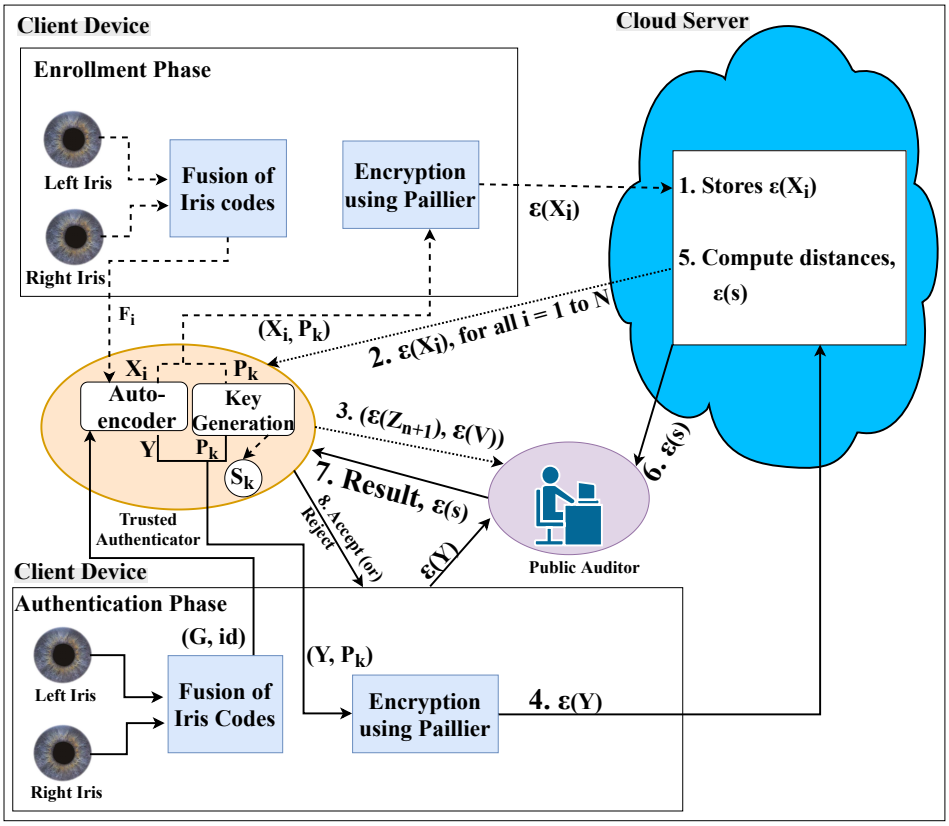


Figure 4.9: Block diagram of SviaPA. The dashed line, Dotted line and Solid line indicates the steps during enrollment, after the enrollment and during the authentication phases.

4.2.1 Preliminaries and Assumptions of SviaPA

4.2.1.1 Autoencoder

Autoencoder is an unsupervised neural network method; which optimizes a rebuilding of the input data in the output layer through a hidden layer with chosen dimensions. Similar to the state-of-the-art dimensionality reduction techniques such as principal component analysis (PCA), linear discriminant analysis (LDA), isometric mapping (ISOMAP), etc., autoencoder can be used to reduce the high-dimensional feature vector [154]. Autoencoder consists of three layers, namely input, hidden and output. The dimensions of input and output layers are the same, whereas the hidden layer contains fewer dimensions. Autoencoder

Algorithm 4.3 Enrollment Phase of SviaPA

Input: Reference left and right iris images of i^{th} user.

1: Client device generates the left and right iris templates from the reference left and right iris images using University of Salzburg tool kit [58].
2: Client device performs the fusion of iris codes, F_i and send F_i to trusted authenticator. *//Refer section 4.1.1.1*
3: Trusted authenticator reduces the dimensions of F_i to improve the performance of the system and send the compressed template, X_i to the client device. *//Refer section 4.2.2*
4: Trusted authenticator generates the public key, P_k and secret key S_k. *//Refer section 4.2.3.1*
5: Client device encrypts X_i and sends the encrypted reference fused compressed iris template, $\varepsilon(X_i)$ to the cloud server. *//Refer section 4.2.3.2*
6: Once all the enrollment phase is completed, the cloud server send $\varepsilon(X_i),\ i\ \in [1,\ N]$ to trusted authenticator.
7: The trusted authenticator generates the encrypted verification vector, $\varepsilon(Z_{n+1})$ using encrypted random vector $\varepsilon(V)$, $\varepsilon(X_i)$ and send ($\varepsilon(Z_{n+1})$, $\varepsilon(V)$) to the public auditor. *//Refer section 4.2.4.2*

consists of two phases, 1) encoder and 2) decoder. An encoder converts the input data into a hidden code, and the decoder reconstructs the original input data from the hidden code. The input and output for an autoencoder are $I \in [0,\ 1]^d$ and $O \in [0,\ 1]^d$, where d is the number of dimensions. Firstly, the encoder maps the input into hidden (or) latent code, $h \in [0,\ 1]^{d'}$, $d' < d$ using the transformation given in equation (4.9).

$$h = S(W \times I + b) \tag{4.9}$$

Where S is a sigmoid function, W is a weight matrix, and b is the bias. By using the decoder, the hidden code, h is then converted back into O with the same dimension as I. The conversion occurs through the transformation given in equation (4.10).

$$O = S(W^{'} \times h + b^{'}) \tag{4.10}$$

Where S is a sigmoid function, $W^{'}$ is a weight matrix of the reverse mapping, and b is the bias. The average reconstruction error is maximized by optimizing the parameters ($W,\ b,\ b^{'}$). The reconstruction error can be measured by either squared error, $L(I, O) = ||I - O||^2$ or binary cross-entropy, $L(I,\ O) = -\sum_{k=1}^{d}[I_k log O_k + (1 - I_k) log(1 - O_k)]$.

Algorithm 4.4 Authentication Phase of SviaPA

Input: Probe left and right iris images, User identifier id of the end user U
Output: Accept or Reject

1: The client device generates the left and right iris templates from the probe left and right iris images using University of Salzburg tool kit [58]. It also acquires the identifier id of the end-user.
2: Client device performs the fusion of iris codes, G and send $(G,\ id)$ to trusted authenticator. *//Refer section 4.1.1.1*
3: Trusted authenticator reduces the dimensions of G to improve the performance of the system and send the fused compressed template, Y to the client device. *//Refer section 4.2.2*
4: The client device encrypts Y and sends the encrypted probe fused compressed iris template, $\varepsilon(Y)$ to the cloud server. *//Refer section 4.2.3.2*
5: The cloud server computes the Manhattan distances, $\varepsilon(d)$ between $\varepsilon(Y)$ & $\varepsilon(X_i)$, $i \in [1,\ N]$ and send $\varepsilon(d)$ to the public auditor. *//Refer section 4.2.4.1*
6: The public auditor checks the correctness of the computed result $\varepsilon(d)$ by using $\varepsilon(Z_{n+1})$, $\varepsilon(V)$, $\varepsilon(Y)$ and sends the verification result to trusted authenticator. *//Refer section 4.2.4.2*
7: If the verification succeeds, then the trusted authenticator considers the Manhattan distance value for the corresponding id given by the end-user to determine whether the user is genuine or not.

To use the autoencoder as a dimensionality reduction technique, use the data obtained in hidden layer and discard the decoder phase.

4.2.1.2 Assumptions

SviaPA assume the following:

- The client device is fully trusted in the enrollment/authentication phase and has limited memory and computational resources.
- The cloud server is malicious as opposed to *Honest-but-curious*.
- The trusted authenticator is a trusted entity which generates the secret and public keys differently for each user. The secret keys of the users are stored securely and broadcast the public keys to the client device.
- The public auditor is only trusted to check the correctness of $\varepsilon(d)$.

4.2.2 Fusion & Reducing the dimensions of Iris code using Autoencoder

This section consists of two phases, namely Fusion and reducing the dimensions of the iris code. In the fusion phase, the iris codes obtained from left and right irises are fused as discussed in section 4.1.1.1. The size of the fused iris code is reduced by using a non-linear dimensionality reduction technique, autoencoder in the reduction phase.

The performance of the system depends on the size of the iris code vector. The fused iris code vector, Z obtained from section 4.1.1.1 is of dimension 1×20480. Reduction in the size improves the overall computational performance of the system. SviaPA uses the autoencoder as a technique to reduce the dimensions of the iris code. Autoencoder is a neural network-based reduction technique and is more efficient than other state-of-the-art linear dimensionality reduction technique such as PCA or non-linear dimensionality reduction techniques such as LDA, ISOMAP, etc [155]. Firstly, the trusted authenticator train the autoencoder using both encoder and decoder phases, but after training, the data obtained after the encoder phase, i.e., in the hidden layer is considered as the reduced feature vector and discard the decoder phase. As the iris code, i.e., the input data to auto-encoder contains 1's and 0's, SviaPA use the cross-entropy as an error function. The 20480-bit binary vector is given as an input to auto-encoder and compressed into 64, 128, 256 and 512-bit respectively.

SviaPA considered 64, 128, 256 and 512 nodes in the hidden layer and computed the EER. Table 4.1 represents the EER values for different sizes of iris code. From Table 4.1, experimentally, we found that there is no loss in the accuracy (EER) if we compress the original iris template of size 20480-bit to 128-bit. Thus, SviaPA reduces the dimensions to 128-bit using the autoencoder and use the 128-bit iris template for further operations.

4.2.3 Ensuring Confidentiality for the Iris templates using Paillier Homomorphic Encryption

Paillier HE [42] is used to ensure the confidentiality of the iris templates during enrollment and authentication phase. The security of the Paillier HE relies on the decisional composite

Table 4.1: EER obtained for databases CASIA-V3-Inerval, IITD and SDUMLA-HMT with different sizes of iris template

	Size of iris code	EER (in %)		
		CASIA-V3-Interval	IITD	SDUMLA-HMT
Compressed Iris template	64-bit	1.38	1.62	0.28
	128-bit	**0.31**	**0.86**	**0.13**
	256-bit	0.43	1.12	0.17
	512-bit	0.47	1.16	0.19
Uncompressed Iris template	**20480-bit**	**0.31**	**0.86**	**0.13**

residuosity assumption (DCRA). HE must satisfy additive property for distance function computation. Paillier HE is an additive homomorphic cryptosystem and is more efficient than other algorithms (e.g., RSA and ElGamal) in terms of encryption and decryption efficiency [42, 153].

Paillier HE scheme consists of four PPT (Probabilistic Polynomial-Time) functions, namely Key Generation (KeyGen), Encryption (Enc), Evaluation (Eval) and Decryption (Dec). The steps involved in each function are explained in the following sections:

4.2.3.1 Key Generation

The function to generate the public key, secret key of Paillier scheme [42] is shown in Figure. 4.10. The function takes two prime numbers m & n as inputs and produces P_k & S_k as output.

(P_k, S_k) ⟵ KeyGen(m, n)

Input: Two large prime numbers, m & n randomly and independently such that gcd(mn, $(m-1)(n-1)$)=1

Output: Public key (P_k) and Secret key (S_k)

- Compute $p = m \cdot n$ and $\lambda = lcm(m, n)$.
- Choose a random integer g, $g \in \mathbb{Z}^*_{p^2}$ such that p divides the order of g.
- Compute $k = I(g^{\lambda} \ (mod\ p^2))$, where function I is defined as $I(u) = \dfrac{(u-1)}{p}$.
- Compute $\phi = k^{-1} \ (mod\ p)$.

$$Public\ key\ (P_k) : [p, g]\ Secret\ key\ (S_k) : [\lambda, \phi].$$

Figure 4.10: Key Generation function in Paillier Homomorphic Encryption

4.2.3.2 Encryption

The function to encrypt the value in Paillier scheme [42] is shown in Figure. 4.11. It takes the plaintext msg and public key P_k as input and produces the encrypted value of msg i.e., $\varepsilon(msg)$ as output.

$\varepsilon(msg)$ ⟵ Enc(P_k, msg)

Input: Public key (P_k), message msg

Output: Encrypted message, $\varepsilon(msg)$

- Choose a random element r such that $gcd(r,\ p) = 1$, $r \in (0,\ p)$ and $r\ \in \mathbb{Z}^*_{\mathfrak{l}}$.
- Compute $\varepsilon(msg) = g^{msg} \cdot r^p \ mod\ p^2$.

Figure 4.11: Encryption function in Paillier Homomorphic Encryption

4.2.3.3 Evaluation

The steps required to perform addition of two original values in Paillier scheme [42] is shown in Figure. 4.12. The addition of two original values can be obtained by the decryption of multiplication of two encrypted values.

$\varepsilon(R) \longleftarrow$ **Eval(**$\varepsilon(a)$**,** $\varepsilon(b)$**,** k**)**

Input: p, encrypted values $\varepsilon(a)$, $\varepsilon(b)$, constant k

Output: Computed result $\varepsilon(R)$

The addition of two values can be computed as

$$\epsilon(R) = add(a, b) = \varepsilon(a) \cdot \varepsilon(b) \; mod \; p^2$$

where r_1, r_2 are the random values considered in encryption function.

The multiplication of the original value with a constant can be computed by raising the encrypted value to the constant.

$$\epsilon(R) = \varepsilon(a)^k \; mod \; p^2$$

Figure 4.12: Evaluation function in Paillier Homomorphic Encryption

4.2.3.4 Decryption

The function to decrypt the encrypted value in Paillier scheme [42] is shown in Figure. 4.13. It takes the encrypted value $\varepsilon(msg)$ and secret key S_k as input and produces the original value msg as output.

$msg \longleftarrow$ **Dec(**S_k**,** $\varepsilon(msg)$**)**
Input: Secret key (S_k), Encrypted message $\varepsilon(msg)$ **Output**: message msg Compute $msg = I(\varepsilon(msg)^{\lambda} \ mod \ p^2) \cdot \phi \ mod \ p$

Figure 4.13: Decryption function in Paillier Homomorphic Encryption

The random number introduced in the encryption process provides the randomness to the encryption result of Paillier. Therefore, Paillier resists chosen plaintext attacks (CPA). The template obtained in section 4.2.2 is encrypted using the Enc function given in Figure. 4.11.

Properties of Paillier HE:

Property 1: Given two encrypted values $\varepsilon(m_1) = Enc(P_k, m_1)$ and $\varepsilon(m_2) = Enc(P_k, m_2)$ for messages m_1 and m_2, decryption of multiplication of the two encrypted values, results in the addition of two original messages and is given in the equation (4.11).

$$D_{S_k}(\varepsilon(m_1) \cdot \varepsilon(m_2) \ mod \ p^2) = m_1 + m_2 \ mod \ p \tag{4.11}$$

Property 2: Given an encrypted value $\varepsilon(m) = Enc(P_k, m)$ for a message m and a constant k, decryption of encrypted value raised to a constant results in the multiplication of the original message and the constant and is given in the equation (4.12).

$$D_{s_k}(\varepsilon(m)^k \ mod \ p^2) = m \cdot k \ \ mod \ p \tag{4.12}$$

4.2.4 Encrypted Distance Computation & Verifying the Correctness of Result

This section describes the computation of Manhattan distance on the encrypted values and a verification procedure to check the correctness of the result returned by the cloud server.

4.2.4.1 Encrypted Distance Computation

SviaPA considered Manhattan distance to compare the reference and probe iris templates. The distance S_{man} = $d_{man}(X, Y)$ can be precisely calculated on the original values by using equation (4.13)

$$S_{man} = |X - Y| \tag{4.13}$$

The cloud server calculates the distance on the encrypted values by using equation (4.11) & equation (4.12) and send to the public auditor.

$$\begin{aligned} S_{man} &= \sum_{i=1}^{M} |X[i] - Y[i]| \\ &= \sum_{i=1}^{M} |X[i] + (-1) \cdot Y[i])| \\ \varepsilon(S_{man}) &= \prod_{i=1}^{M} \varepsilon(X[i]) \cdot \varepsilon(Y[i])^{-1} \; mod \; p^2 \end{aligned}$$

$$\varepsilon(S_{man}) = \prod_{i=1}^{M} \varepsilon(X[i]) \cdot \varepsilon(Y[i])^{-1} \; mod \; p^2 \tag{4.14}$$

$\varepsilon(Y)^{-1}$ represents the multiplicative inverse of $\varepsilon(Y)$ in the integers modulo p. If the vector Y contains smaller values than X, then after decryption the correct subtraction result will come. On the other hand, if the vector Y contains larger values than X, then after decryption, the result lies in between 0 and p-1. The obtained result will be subtracted from p to get the correct subtraction result. For example, if the result is -1, then we will get p-1 as a result after decryption. So, to get -1 as a result, return p-1-p as a result after decryption.

4.2.4.2 Verifying the Correctness of Result

The distances between X_i and Y are given by

$$d = \{r_i / r_i = \sum_{j=1}^{M} (X_i[j] + Y[j]), \forall \; i = 1 \; to \; N\} \tag{4.15}$$

$Y[j]$ is the multiplicative inverse integers in the modulo p. The cloud server computes the Manhattan distances on the encrypted values, $\varepsilon(d) = \varepsilon(r_i), \forall\, i = 1\ to\ N$ between $\varepsilon(X_i)\ \forall\ i = 1\ to\ N$ and $\varepsilon(Y)$ by using the properties of Paillier. The distances on the encrypted values are given in equation (4.16).

$$\varepsilon(d) = \{\varepsilon(r_i)/\varepsilon(r_i) = \prod_{j=1}^{M}(\varepsilon(X_i[j]) \cdot \varepsilon(Y[j])), \forall\, i = 1\ to\ N\} \tag{4.16}$$

Since both the reference and the probe templates are in encrypted form, the privacy of iris templates, i.e., user privacy is maintained. The verification scheme allows the public auditor to check the correctness of $\varepsilon(d)$ returned by the cloud server.

Generation of encrypted verification vector ($\varepsilon(Z_{n+1})$): After the enrollment phase, the trusted authenticator constructs the encrypted verification vector using $\varepsilon(X_i), \forall\, i = 1\ to\ N$ returned by the cloud server. The encrypted verification vector helps the public auditor to check the correctness of the Manhattan distances. Let $\varepsilon(Z_{n+1})$ be the encrypted verification vector and is computed on the encrypted values by using equation (4.11).

$$\begin{aligned} Z_{n+1}[j] &= \sum_{i=1}^{N}(X_i[j] + v_i),\ \forall\, j = 1\ to\ M \\ \varepsilon(Z_{n+1}[j]) &= \prod_{i=1}^{N}(\varepsilon(X_i[j]) \cdot \varepsilon(v_i)),\ \forall\, j = 1\ to\ M \end{aligned} \tag{4.17}$$

where, $v_i\ \forall\ i = 1\ to\ N$ are the random integers, and $\varepsilon(V) = (\varepsilon(v_1), \varepsilon(v_2), ..., \varepsilon(v_N))$. As long as the secret key is secure, encrypted verification vector is also secure and its security relies on the hardness of DCRA. The steps involved in the generation of encrypted verification vector are given in Algorithm 4.5. The trusted authenticator implements the Algorithm 4.5 after the training phase. The verification vector denoted as Z_{n+1} with same dimension of X_i is initialized to (1, 1, ..., 1). Encrypt Z_{n+1} using the public key P_k. The function *randomInteger()* generates a random value v_i. Encrypt v_i using the public key P_k. The random value generated in each and every iteration is encrypted with different public keys. The keys used to encrypt v_i are completely different from the keys used to encrypt iris

Algorithm 4.5 Generation of Encrypted Verification Vector in SviaPA

Input: $\varepsilon(X_1), \varepsilon(X_2), \ldots, \varepsilon(X_N)$,
Output: $\varepsilon(Z_{n+1}), \varepsilon(V)$

begin
  $Z_{n+1} = (1, 1, ..., 1)$
  for $i \leftarrow 1$ `to` N **do**
    $v_i \leftarrow randomInteger()$
    $\varepsilon(v_i) \leftarrow Enc(P_k, v_i)$
  end for
  for $j \leftarrow 1$ `to` M **do**
    $\varepsilon(Z_{n+1}[j]) \leftarrow Enc(P_k, Z_{n+1}[j])$
    for $i \leftarrow 1$ `to` N **do**
      $\varepsilon(tmp) \leftarrow multiply(\varepsilon(X_i[j]), \varepsilon(v_i))$
      $\varepsilon(Z_{n+1}[j]) \leftarrow multiply(\varepsilon(Z_{n+1}[j]), tmp)$
    end for
  end for
  $\varepsilon(Z_{n+1}) = (\varepsilon(Z_{n+1}[1]), \varepsilon(Z_{n+1}[2]), ..., \varepsilon(Z_{n+1}[M]))$
  $\varepsilon(V) = (\varepsilon(v_1), \varepsilon(v_2), ..., \varepsilon(v_N))$
return $(\varepsilon(Z_{n+1}), \varepsilon(V))$
end

templates by client device. *multiply* function is used to achieve the property 1 of Paillier.

The function *multiply* is called to perform the multiplication between j^{th} value of encrypted reference template $\varepsilon(X_i)$ and encrypted random value $\varepsilon(v_i)$, where i varies from 1 to N. $\varepsilon(tmp)$ stores the multiplication result. The function *multiply* is called to perform the multiplication between $\varepsilon(Z_{n+1})$ and $\varepsilon(tmp)$. After the completion of M iterations, the encrypted verification vector $\varepsilon(Z_{n+1})$ which is shown in equation (4.17) is obtained. The N random values are assigned to $\varepsilon(V)$. After the enrollment phase, the trusted authenticator send $\varepsilon(Z_{n+1})$ and $\varepsilon(V)$ to the public auditor.

Ensuring the correctness of Manhattan distance: The public auditor checks the correctness of Manhattan distances $\varepsilon(d)$ using the $\varepsilon(Z_{n+1})$, $\varepsilon(Y)$ and $\varepsilon(V)$. The verification scheme checks the correctness of the result on the encrypted values itself; as a result, anyone can perform the correctness of the $\varepsilon(d)$ without the private information of the user. The steps involved to check the correctness of Manhattan distances are described in Algorithm 4.6.

Algorithm 4.6 Correctness of result in SviaPA

Input: $\varepsilon(Z_{n+1}), \varepsilon(V), \varepsilon(Y), \varepsilon(d)$
Output: Zero or Non-Zero

1: **begin**
2: $D1 = 1$
3: $\varepsilon(D1) \leftarrow Enc(P_k, D1)$
4: **for** $j \leftarrow 1$ to M **do**
5: $\varepsilon(tmp_j) \leftarrow mul\_const(\varepsilon(Y[j]), N)$
6: $\varepsilon(temp_j) \leftarrow multiply(\varepsilon(Z_{n+1}[j]), \varepsilon(tmp_j))$
7: $\varepsilon(D1) \leftarrow multiply(\varepsilon(D1), \varepsilon(temp_j))$
8: **end for**
9: $D2 = 1$
10: $\varepsilon(D2) \leftarrow Enc(P_k, D2)$
11: **for** $i \leftarrow 1$ to N **do**
12: $\varepsilon(t_i) \leftarrow mul\_const(\varepsilon(v_i), M)$
13: $\varepsilon(te_i) \leftarrow multiply(\varepsilon(r_i), \varepsilon(t_i))$
14: $\varepsilon(D2) \leftarrow multiply(\varepsilon(D2), \varepsilon(te_i))$
15: **end for**
16:
17: $D1 \leftarrow Dec(S_k, \varepsilon(D1))$
18: $D2 \leftarrow Dec(S_k, \varepsilon(D2))$
19: **return** $(\varepsilon(D1) - \varepsilon(D2))$
20: **end**

The steps (4-8) of Algorithm 4.6 computes $\varepsilon(D1) = \prod_{j=1}^{M}(\varepsilon(Z_{n+1}[j]) \cdot \varepsilon(Y[j])^N)$. The steps (11-15) of Algorithm 4.6 computes $\varepsilon(D2) = \prod_{i=1}^{N}(\varepsilon(r_i) \cdot \varepsilon(v_i)^M)$. The public auditor decrypt $\varepsilon(D1)$ and $\varepsilon(D2)$. The keys used to encrypt/decrypt $D1$ & $D2$ by public auditor are completely different from the keys used to encrypt iris templates by client device. mul_const is called to perform encrypted value raised to a constant value which results in multiplication of constant and corresponding plaintext value. Finally, compute $D1 - D2$ and send the result to trusted authenticator. If the result is a zero value, the Manhattan distances $\varepsilon(d)$ returned by the cloud server are considered to be correct. For better clarity, First, we prove that the $D1$ and $D2$ are same in the normal domain by using the equation (4.16), equation (4.17) and some algebraic properties of vectors. Later, we use the properties of Paillier, i.e., equation (4.11) and equation (4.12) to achieve the same on the encrypted values. If the verification succeeds then the Manhattan distances $\varepsilon(d)$ is considered correct. So, the trusted authenticator finds the value with index id given by the

end-user from $\varepsilon(d)$. The value is compared with a threshold, τ to determine whether the user is genuine or not.

The time required to compute the N Manhattan distances between iris templates X_i and Y, each of dimension M is $\mathcal{O}(NM)$. We excluded the time required to compute Z_{n+1} as it is computed only once after the enrollment phase. The time required to compute $D1$ and $D2$ are $\mathcal{O}(M)$ and $\mathcal{O}(N)$. The total time required for verification of computed result returned by the cloud server is $\mathcal{O}(N+M)$, which is less than the time required to compute the distances. The steps to check whether $D1$ and $D2$ are given below.

$$
\begin{aligned}
D1 &= \sum_{j=1}^{M}(Z_{n+1}[j]) + NY[j]) \\
&= \sum_{j=1}^{M}(\sum_{i=1}^{N}(X_i[j] + v_i) + NY[j]) \; //Using\ equation(4.17) \\
&= \sum_{j=1}^{M}(\sum_{i=1}^{N} X_i[j] + \sum_{i=1}^{N} v_i + NY[j]) \\
&= \sum_{j=1}^{M}\sum_{i=1}^{N} X_i[j] + \sum_{j=1}^{M}\sum_{i=1}^{N} v_i + \sum_{j=1}^{M}\sum_{i=1}^{N} Y[j] \\
&= \sum_{i=1}^{N}(\sum_{j=1}^{M} X_i[j] + \sum_{j=1}^{M} Y[j] + \sum_{j=1}^{M} v_i) \\
&= \sum_{i=1}^{N}(\sum_{j=1}^{M}(X_i[j] + Y[j]) + Mv_i) \\
&= \sum_{i=1}^{N}(r_i + Mv_i) \quad //Using\ equation(4.15) \\
&= D2
\end{aligned}
$$

$D1$ and $D2$ can be realized on the encrypted values by using equation (4.11) and (4.12) as follows.

$$
\begin{aligned}
\varepsilon(D1) &= \prod_{j=1}^{M}(\varepsilon(Z_{n+1}[j])) \cdot \varepsilon(Y[j])^N) \\
\varepsilon(D2) &= \prod_{i=1}^{N}(\varepsilon(r_i)) \cdot \varepsilon(v_i)^M)
\end{aligned}
$$

The proof to check $\varepsilon(D1) = \varepsilon(D2)$ are similar to the steps in the normal domain.

4.2.5 Limitations of SviaPA

In SviaPA, we introduce a public auditor as a third party to check the correctness of result returned by the cloud server. The limitations of SviaPA are as follows:

- SviaPA assume that the public auditor is Honest-but-Curious, means the public auditor follows the verification protocol honestly but curious to know the information. Our verification scheme performs on the encrypted data; as a result, it is difficult for the public auditor to know the original data. However, if the public auditor is a malicious entity and returns an incorrect verification result, then the imposter may get access into the system.
- The trusted authenticator compute $\varepsilon(Z_{n+1})$ by using $\varepsilon(X_i)$, as a result the number of users are fixed in SviaPA. If a new user wants to authenticate using SviaPA, then $\varepsilon(Z_{n+1})$ need to be recomputed by including the new user template.
- SviaPA require extra time to verify the correctness of the Manhattan distances returned by the cloud server.

4.3 SviaB: Secure and Verifiable Multi-Instance Iris Authentication using Blockchain

SviaB leverage emerging technologies like Blockchain and smart contract to overcome the limitations of SviaPA discussed in section 4.2.5. Blockchain has been developed to allow decentralized consensus between two non-trusting agents. Autoencoder is used in SviaB as a dimensionality reduction technique to overcome the limitations of BMIAE discussed in section 4.1.5. SviaB is the first multi-instance iris authentication system to combine Paillier HE and Blockchain technology to achieve privacy and integrity against malicious computing server. SviaB focuses on achieving both the confidentiality of fused iris templates and integrity of fused reference iris templates as well as the trust of the matching result. The

privacy of iris templates is achieved by encrypting the iris templates using Paillier HE [42]. The smart contract running on a Blockchain network computes the distance between the encrypted reference template and encrypted probe template; hence, the integrity of computation is achieved.

Assumptions of SviaB:

SviaB assume the following:

- During the enrollment/authentication phase, the client device is fully trusted and has limited memory and computational resources.
- The trusted authenticator is a trusted entity which generates the secret and public keys differently for each user. The secret keys of the users are stored securely and broadcast the public keys to the client device.
- The server & client device need not store the entire ledger of the Blockchain network.
- The consensus algorithm of the Blockchain is secure & robust against security attacks of the Blockchain.
- The contract address of the smart contract is shared with the server & the client device prior to the enrollment phase.

Since the data present in the Blockchain is visible to all the nodes present in the Blockchain, privacy problem may exist. To overcome this limitation, SviaB encrypts the fused templates using Paillier HE before sending to the Blockchain. As long as the secret key used to decrypt the template is secure, even if the encrypted templates are exposed, SviaB is secure due to the hardness of computation of DCRA. In Blockchain, the storage cost is expensive when compared to computation [119]. To overcome this limitation, $SviaB$ stores only the hash value of the encrypted reference templates in the Blockchain and stores the encrypted reference templates in the server itself.

The flow diagram for SviaB is shown in Figure. 4.14. SviaB consists of two phases, namely enrollment phase and authentication phase whose participants are a client device, a centralized server and a Blockchain network. The steps involved in the enrollment and

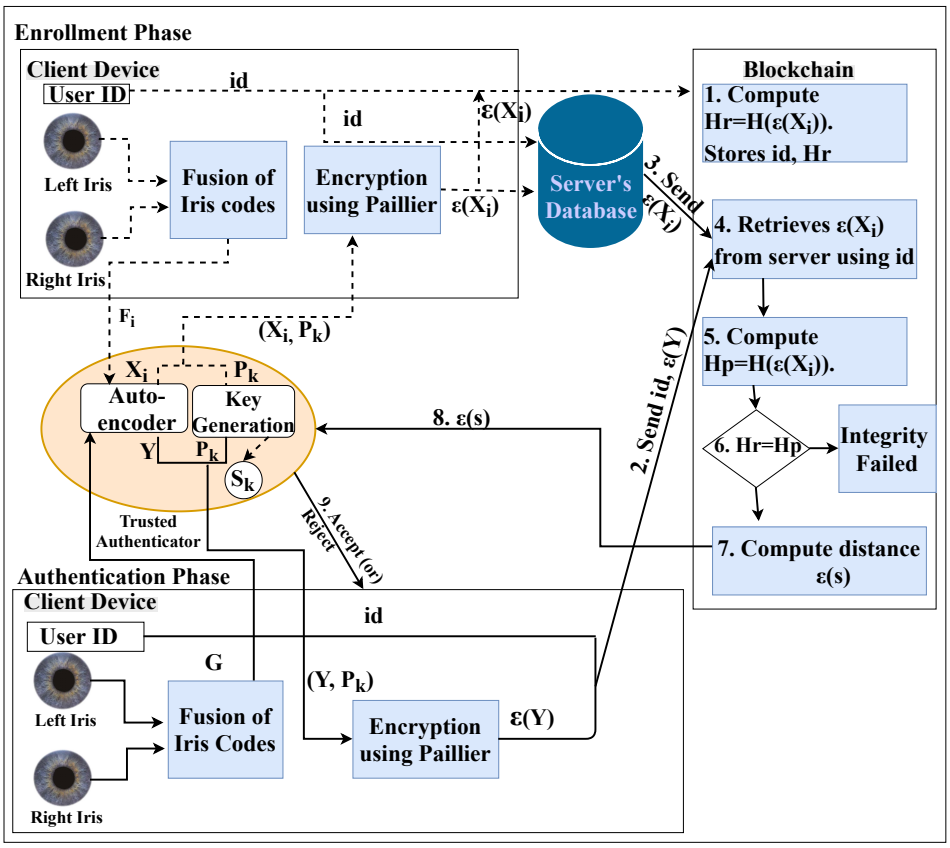


Figure 4.14: Block diagram of SviaB. The dashed line and Solid line indicates the steps during enrollment and the authentication phases.

authentication phases for SviaB are described in Algorithm 4.7 and Algorithm 4.8. The phases like Fusion, Reducing the dimensions of the iris template and ensuring the confidentiality for the iris templates are the same as SviaPA. The formal smart contract to achieve the integrity of encrypted reference iris template and trust on the computed distance is shown in Figure. 4.8.

4.4 Implementation details and Security Analysis

The following measures are used to evaluate the efficiency of a biometric system according to biometric information protection [23].

1. Performance evaluation in terms of EER, d' and KS-test.
2. Irreversibility and Unlinkability Analysis.

Algorithm 4.7 Enrollment Phase of SviaB

Input: Reference left and right iris images of i^{th} user.

1: Trusted authenticator generate the public key, P_k and secret key S_k.
2: Client device generates the left & right iris templates from the reference left & right iris images using University of Salzburg tool kit [58]. It also acquires the corresponding user identifier, id of the end user U.
3: Client device performs the fusion of iris codes, F_i and send F_i to trusted authenticator. *//Refer section 4.1.1.1*
4: Trusted authenticator reduces the dimensions of F_i to improve the performance of the system and send the reference fused compressed template, X_i to the client device. *//Refer section 4.2.2*
5: Client device encrypts X_i and sends the encrypted reference fused compressed iris template, $\varepsilon(X_i)$ along with id to cloud server and Blockchain. *//Refer section 4.2.3.2*
6: The Blockchain computes the hash value of $\varepsilon(X_i)$, Hr and stores Hr along with id. (Since the storage in the Blockchain is more expensive than computation, SviaB stores the hash value of encrypted reference templates in Blockchain and encrypted reference templates in the server.)

Algorithm 4.8 Authentication Phase of SviaB

Input: Probe left and right iris images, User identifier id of the end user U
Output: Accept or Reject

1: Client device generates the left & right iris templates from the probe left & right iris images using University of Salzburg tool kit [58]. It also acquires the identifier, id of the end-user.
2: Client device performs the fusion of iris codes, G and send G to trusted authenticator. *//Refer section 4.1.1.1*
3: Trusted authenticator reduces the dimensions of G to improve the performance of the system and send the compressed template, Y to the client device. *//Refer section 4.2.2*
4: Client device encrypts Y and sends the encrypted probe fused compressed iris template, $\varepsilon(Y)$ along with id to the Blockchain. *//Refer section 4.2.3.2*
5: The Blockchain retrieves $\varepsilon(X_i)$ from server using id.
6: The Blockchain computes the hash value of retrieved $\varepsilon(X_i)$, Hp.
7: The Blockchain compares Hr and Hp. If the hash values are differ then the Blockchain will inform to the client device that the server database is compromised otherwise it computes the distance, $\varepsilon(s)$ and send to the trusted authenticator.
8: The trusted authenticator decrypts the distance, $\varepsilon(s)$ using the secret key S_k and obtains R. The trusted authenticator compares R with τ and decides whether the user is genuine or not.

3. Computational cost in terms of time taken to perform operations.

4.4.1 Performance Evaluation of BMIAE, SviaPA and SviaB

The EER obtained for only left iris (OLI), only right iris (ORI), fused iris (FT) and fused compressed iris template (FCT) in unprotected and protected systems for BMIAE, SviaPA and SviaPB are shown in Table 4.2 & Table 4.3. We observe that there is no loss of accuracy in the protected system from Table 4.2 & Table 4.3.

Table 4.2: EER obtained in unprotected system for BMIAE, SviaPA and SviaB

Database	OLI	ORI	FT	FCT		
				BMIAE	SviaPA	SviaB
CASIA-V3-Interval	3.26	4.41	0.31	0.13	0.31	0.31
IITD	4.41	4.15	0.86	0.88	0.86	0.86
SDUMLA-HMT	2.10	1.28	0.13	0.0002	0.13	0.13

Table 4.3: EER obtained in protected system for BMIAE, SviaPA and SviaB

Database	OLI	ORI	FT	FCT		
				BMIAE	SviaPA	SviaB
CASIA-V3-Interval	3.26	4.41	0.31	0.13	0.31	0.31
IITD	4.41	4.15	0.86	0.88	0.86	0.86
SDUMLA-HMT	2.10	1.28	0.13	0.0002	0.13	0.13

The baseline comparison of EER, storage cost and time for BMIAE, SviaPA & SviaB are shown in Table 4.4, Table 4.5. The unprotected and uncompressed template (UUT) indicates the template without compression and encryption, compressed and unprotected template (CUT) indicates the template with compression and without encryption, and compressed and protected template (CPT) indicates the template with compression and encryption. We can infer from Table 4.4 and Table 4.5 that there is no degradation of accuracy with BMIAE, SviaPA/SviaB. The same EER can be obtained if the distance is computed either in the Blockchain or in a server. So, the DET curves, EER, d′, & KS-test values are same for SviaPA and SviaB.

Table 4.4: Baseline Comparison in terms of Storage cost (in Kilo Bytes (KB)), EER and Time (Average in seconds) for BMIAE

Database	Template type	Template size	Storage cost	EER (%)	Time
CASIA-V3-Interval	UUT and distance computation in server	20480	228	0.31	0.035
	CUT and distance computation in server	1280	25	0.13	0.009
	CPT and distance computation in server	1280	25	0.13	0.062
	CPT and distance computation in Blockchain	1280	-	0.13	6.0254
IITD	UUT and distance computation in server	20480	413	0.86	0.054
	CUT and distance computation in server	2560	88	0.88	0.023
	CPT and distance computation in server	2560	88	0.88	0.12729
	CPT and distance computation in Blockchain	2560	-	0.88	10.0472
SDUMLA-HMT	UUT and distance computation in server	20480	207.5	0.13	0.048
	CUT and distance computation in server	2560	23	0.0002	0.021
	CPT and distance computation in server	2560	23	0.0002	0.1163
	CPT and distance computation in Blockchain	2560	-	0.0002	10.0462

The DET curves of BMIAE for different databases are shown in Figure. 4.15. The separability measures (d' & KS-test values) and EER on encrypted data of BMIAE for different databases are shown in Figure. 4.16. The DET curves of SviaPA/SviaB for different databases are shown in Figure. 4.17. The separability measures (d' & KS-test values) and EER on encrypted data of SviaPA/SviaB for different databases are shown in Figure. 4.18. The clear separation between genuine and imposter scores of BMIAE and SviaPA/SviaB for different databases are shown in Figure. 4.19 and Figure. 4.20.

Table 4.5: Baseline Comparison in terms of Storage cost (in Kilo Bytes (KB)), EER and Time (Average in seconds) for SviaPA and SviaB

Database	Template type	Template size	Storage cost	EER (%)	Time
CASIA-V3-Interval	UUT and distance computation in server	20480	228	0.31	0.035
	CUT and distance computation in server	128	12	0.31	0.0094
	CPT and distance computation in server (SviaPA)	128	58	0.31	0.33
	CPT and distance computation in Blockchain (SviaB)	128	-	0.31	1.33
IITD	UUT and distance computation in server	20480	413	0.86	0.054
	CUT and distance computation in server	128	88	0.88	0.094
	CPT and distance computation in server (SviaPA)	128	88	0.86	0.33
	CPT and distance computation in Blockchain (SviaB)	128	-	0.86	1.33
SDUMLA-HMT	UUT and distance computation in server	20480	207.5	0.13	0.048
	CUT and distance computation in server	128	10.5	0.13	0.0094
	CPT and distance computation in server (SviaPA)	128	40	0.13	0.33
	CPT and distance computation in Blockchain (SviaB)	128	-	0.13	1.33

4.4.2 Security Analysis of BMIAE, SviaPA and SviaB

The template protection method must satisfy the requirements of irreversibility, revocability and unlinkability to ensure the privacy of the iris templates. The vulnerability of attacks in BMIAE can occur in the following entries:

1. The server.

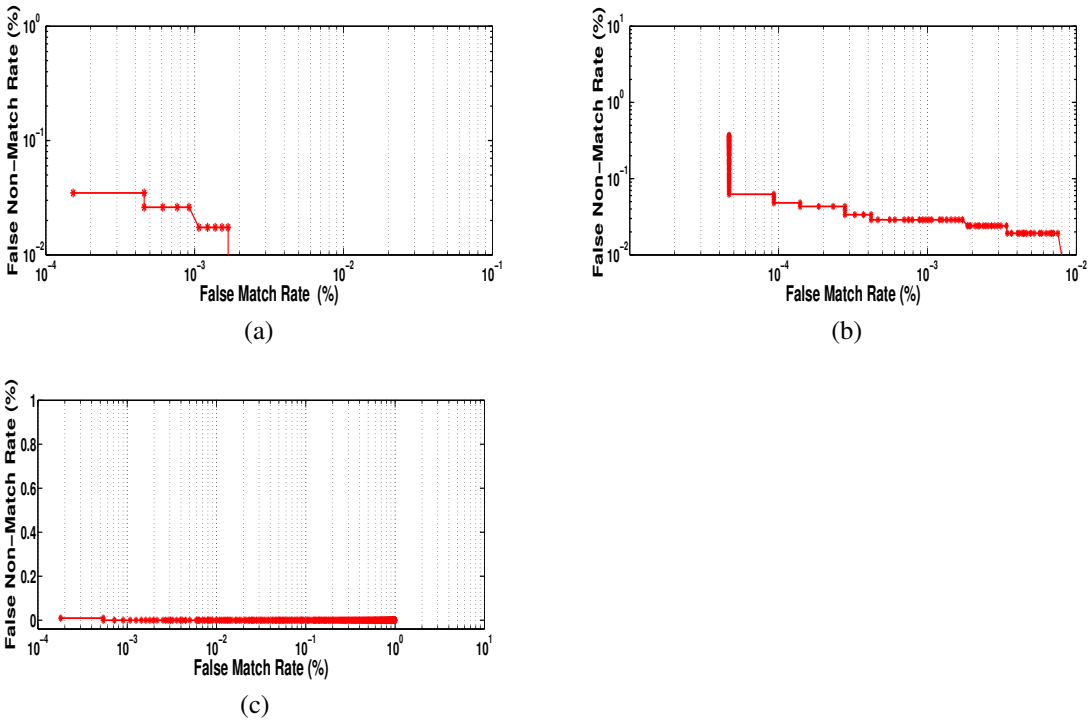


Figure 4.15: DET curves of BMIAE for (a) CASIA-V3-Interval, (b) IITD, (c) SDUMLA-HMT databases

2. The client device.
3. The communication channel between the server/Blockchain network and the client device.
4. Blockchain network

In BMIAE, the client device extracts the features of the iris image, and the secret key is also stored in the client device. Hence, security is to be ensured for the client device. As, we assume the client device is a trusted entity, the keys and features of iris image are secure. The server only stores the templates which are encrypted using ElGamal and the hash value of the encrypted templates is stored in smart contract. Since the security of ElGamal depends on the apparent hardness of solving the discrete logarithm problem on a cyclic group, the iris templates stored in the server database are secure. It isn't easy to decrypt the encrypted iris templates without the secret key. As a result, the communication channel is also reliable.

The vulnerability of attacks in SviaPA and SviaB can occur in the following entries:

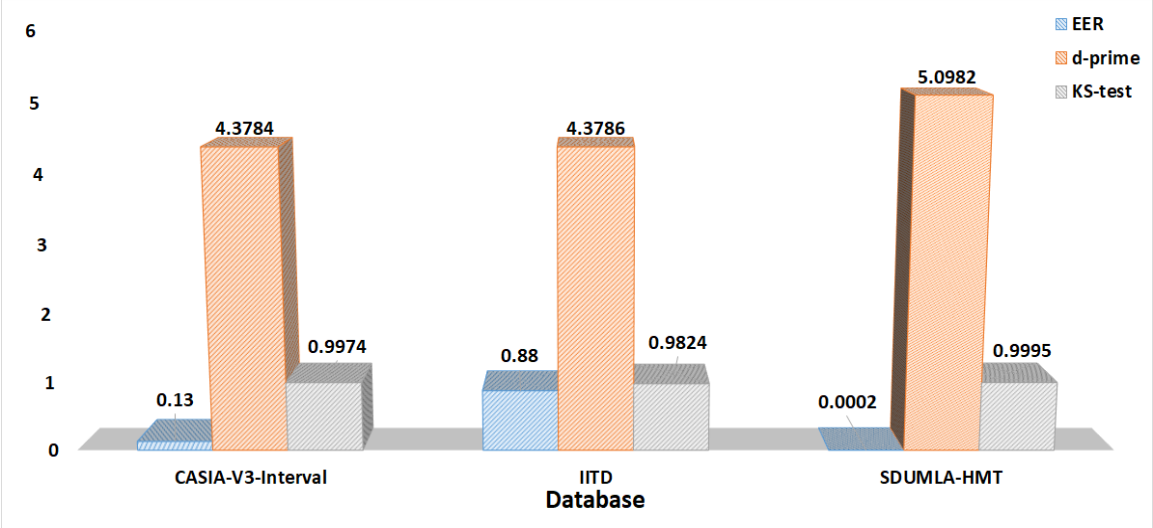


Figure 4.16: EER, Separability Measures (d′ and KS test) of BMIAE for CASIA-V3-Interval, IITD and SDUMLA-HMT databases

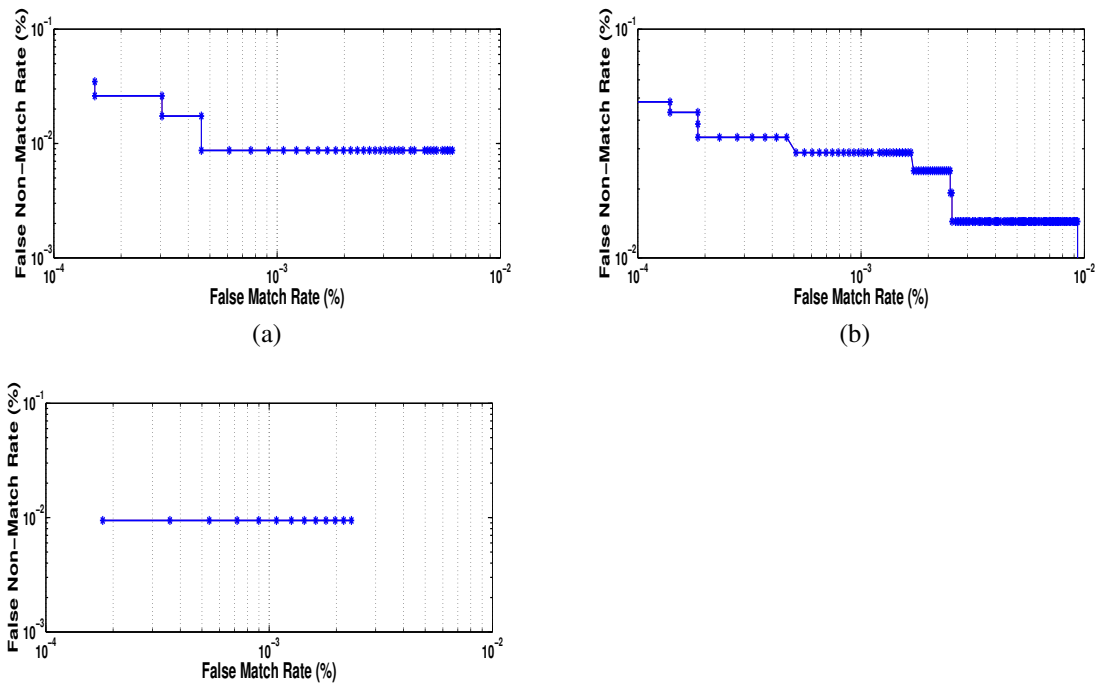


Figure 4.17: DET curves of SviaPA or SviaB for (a) CASIA-V3-Interval, (b) IITD, (c) SDUMLA-HMT databases

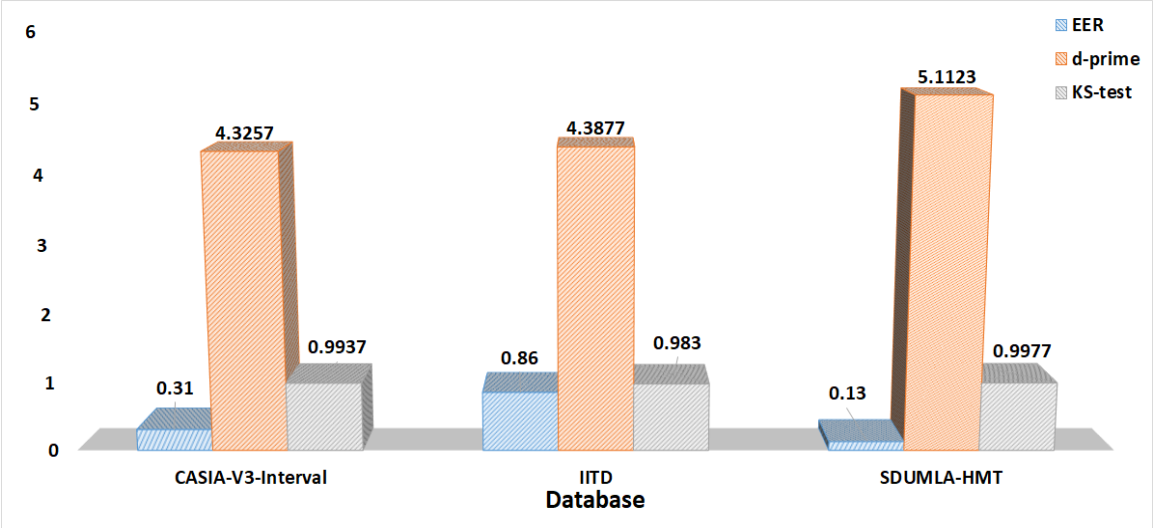
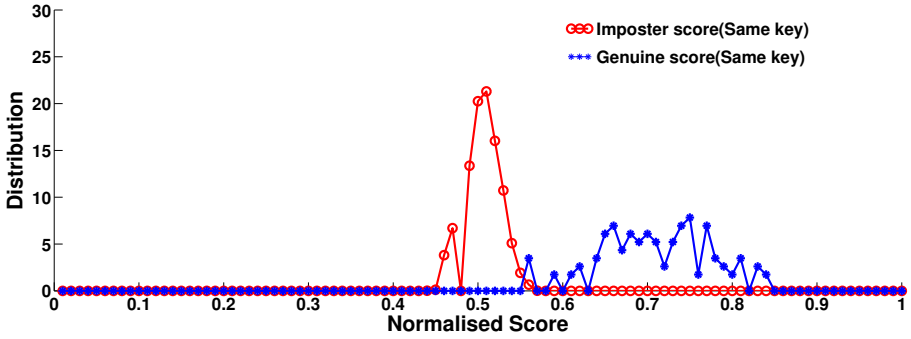


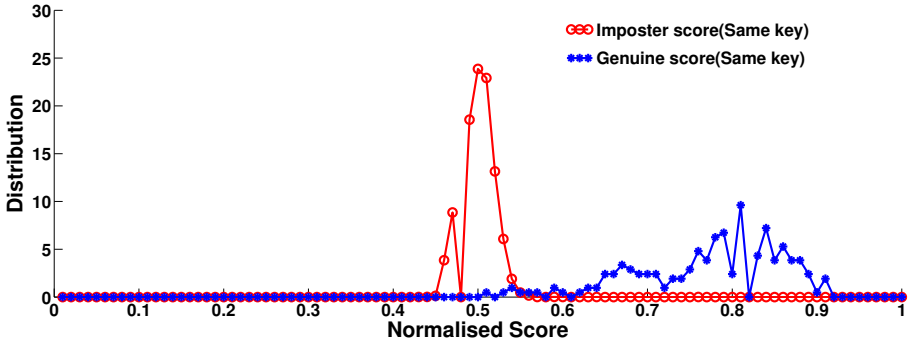
Figure 4.18: EER, Separability Measures (d′ and KS test) of SviaPA or SviaB for CASIA-V3-Interval, IITD and SDUMLA-HMT databases

1. The cloud server.
2. The client device.
3. The communication channel between the server and the client device.
4. Blockchain network.
5. The trusted authenticator
6. The public auditor.

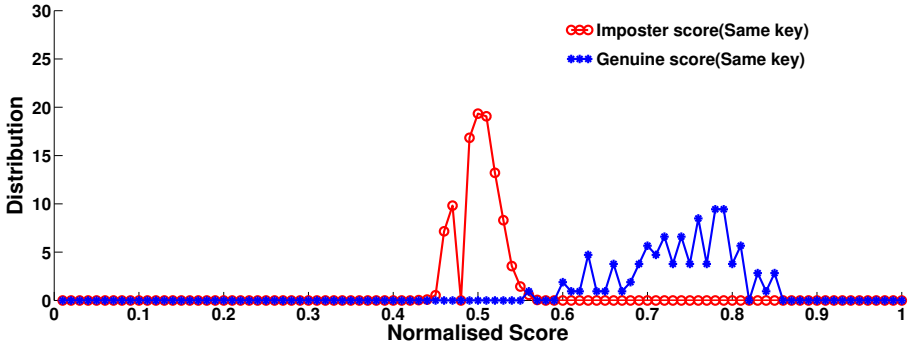
In SviaPA/SviaB, the client device extracts the features from the iris image. Hence, security is to be ensured for the client device. SviaPA/SviaB assume that the client device is a trusted entity. The trusted authenticator generates the keys needed for encryption and decryption. SviaPA and SviaB assume that the trusted authenticator is also a trusted entity. In SviaB, the server only stores the templates which are encrypted using Paillier and the smart contract only stores the hash of the encrypted templates. The security of the Paillier HE relies on the hardness of solving the decisional composite residuosity assumption (DCRA). The data is always secure even if an attacker attacks the communication channel because encrypted iris template cannot be decrypted without a secret key. The encrypted iris templates in the cloud server are secure since the security of both SviaPA and SviaB depends on the



(a)

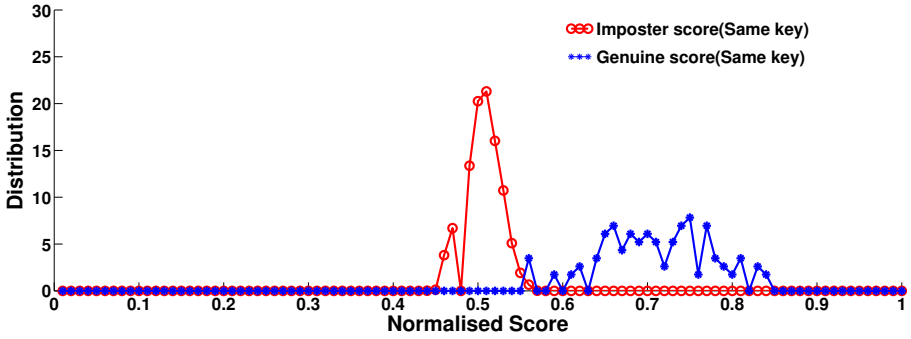


(b)

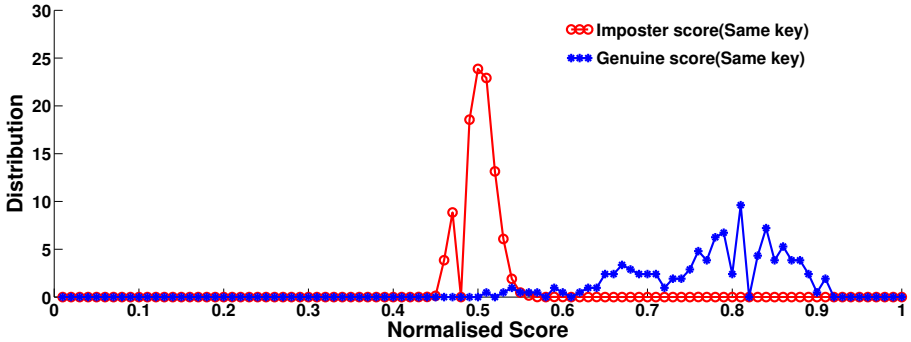


(c)

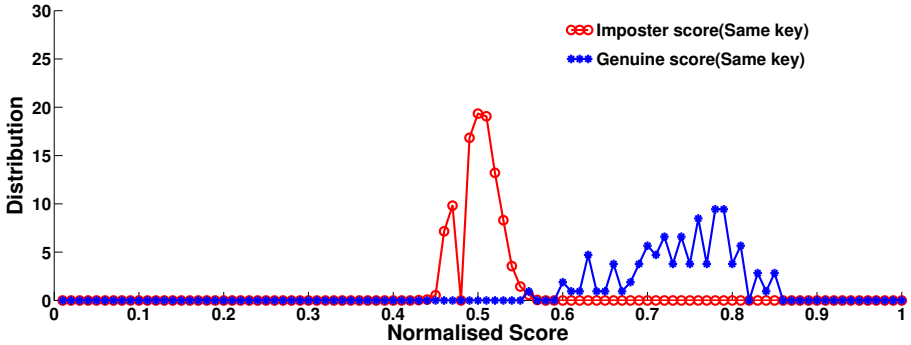
Figure 4.19: Genuine and Imposter distributions of BMIAE for (a) CASIA-V3-Interval (b) IITD and (c) SDUMLA-HMT databases



(a)



(b)



(c)

Figure 4.20: Genuine and Imposter distributions of SviaPA/SviaB for (a) CASIA-V3-Interval (b) IITD and (c) SDUMLA-HMT databases

DCRA. The public auditor in SviaPA verifies the computation result without using the private information of the user.

Irreversibiltiy Analysis: Irreversibility refers to obtaining the original template from the encrypted template.

In BMIAE and SviaB, the client device sends the encrypted reference template of a user to the server and the smart contract during the enrollment phase. The encrypted reference template is stored in the server wheres hash value of the encrypted reference template is stored in the smart contract. The client device sends the encrypted probe template to the Blockchain during the authentication phase. The smart contract retrieves the encrypted reference template from the server and computes the distance between encrypted reference and encrypted probe iris templates. The smart contract sends the computed encrypted result to the client device. Only the client device has the secret key (S_k) to decrypt the result. As the BMIAE uses ElGamal HE scheme [39] to protect the templates, and the security of ElGamal scheme relies on solving the discrete logarithm problem, it is computationally infeasible to decrypt the templates by the server or an imposter without secret key (S_k). Therefore, BMIAE satisfies the irreversibility property. As the SviaB uses Paillier HE scheme [42] to protect the templates, and the security of Paillier scheme relies on solving the DCRA, it is computationally infeasible to decrypt the templates by the server or an imposter without secret key (S_k). Therefore, SviaB satisfies the irreversibility property.

In SviaPA, during the enrollment phase, the cloud server stores the encrypted reference templates. During the authentication phase, the client device sends the encrypted probe template to the server to calculate the distances. The server computes the Manhattan distances and sends the computed encrypted result to the public auditor. The trusted authenticator sends the $\varepsilon(Z_{n+1})$ and $\varepsilon(V)$ to the public auditor; as a result, the reference templates cannot be accessed by the auditor. The public auditor checks the correctness of the computed result without using the secret information of the user. The trusted authenticator can decrypt the result with the secret key (S_k). As mentioned earlier, it is computationally infeasible to decrypt the templates without secret key (S_k). The security of SviaPA depends on solving the DCRA, which is an NP-Hard. Hence SviaPA satisfies the property of irreversibility requirement standards.

Revocability: Revocability ensures that a new protected template should be generated by the protection method if the old template is compromised or stolen. In BMIAE, SviaPA & SviaB, Revocability can be achieved by re-encrypting the samples in the database with a new key pair (P'_k, S'_k) instead of acquiring the new samples from the users.

Unlinkability: Unlinkability ensures that there won't be any correlation between the protected templates used in different applications. Elgamal, Paillier schemes used in BMIAE, SviaPA/SviaB are based on probabilistic encryption. Due to the randomness involved in bothe ElGamal & Paillier schemes, different ciphertexts can be generated even if the same message is encrypted multiple times with the same key, and there won't exist any similarity between the generated ciphertexts.

4.4.3 Computational Analysis of BMIAE, SviaPA and SviaB

The computational cost in terms of time, cost and number of the operations is discussed in the following sections.

4.4.3.1 Computational cost in terms of time & cost

The time required to perform the encryption/decryption of BMIAE, SviaPA/SviaB on different databases is shown in Table 4.6. The computation cost & time required to execute operations in a smart contract in units of gas & dollars and in units of seconds are shown in Table 4.7. The reduced iris code size is the same for all considered databases in SviaPA/SviaB, whereas the iris code size varies for each database in BMIAE to obtain optimal accuracy. Therefore, in Table 4.6, the encryption/decryption time is same for all databases in SviaPA/SviaB. The comparison of time to compute the distance in the Blockchain and the server is illustrated in Table 4.4 & Table 4.5. The increase in the computation provides an enhanced functionality (i.e., trust on the computed distance without any third party) to SviaB & BMIAE.

Table 4.6: Computational cost (for encryption and decryption (Average in secs)

Method	Database	Template size	Encryption	Decryption
BMIAE	CASIA-V3-Interval	1280	0.0184	0.007
	IITD	2560	0.03521	0.012
	SDUMLA-HMT	2560	0.03519	0.011
SviaPA/SviaB	-	128	0.00242	0.00001

4.4.3.2 Computational cost in terms of number of operations

The privacy of the fused reference and probe iris templates in BMIAE, SviaPA and SviaB are ensured by performing the encryption using P_k before sending to the server/Blockchain. The client device or trusted authenticator needs to perform only one encryption and decryption in BMIAE or SviaPA/SviaB. The number of exponentiation, multiplications, and encryptions/decryptions required in BMIAE, SviaPA and SviaB for different databases are shown in Table 4.8.

- As, BMIAE and SviaB are verification systems, they need to compute the distance between the probe & corresponding reference template associated with id only.
- SviaPA need to compute N Manhattan distances between the probe and each reference template. So, a single distance is multiplied by the number of reference templates. We include the computational cost of distance computation only and exclude the cost required to check the correctness of the result.

4.4.4 Comparison Analysis of BMIAE, SviaPA and SviaB with existing methods

The EER comparison of BMIAE, SviaPA/SviaB with state-of-the-art works is shown in Table 4.9. We can infer that BMIAE, SviaPA/SviaB shows better EER value when compared

Table 4.7: Computation cost and time taken by smart contract to perform each operation. (BMIAE and SviaB considered a gas price of 3 gwei (1 gwei = 10^{-9} ETH) and 1 ETH = $176.83 for BMIAE (Real world values at time of writing, September 03, 2019) and 1 ETH = $239.15 for SviaB (Real world values at time of writing, March 07, 2020)

Method	Database	templ-ate size	Operation							
			Smart Contract Deployment		Computation of Hash and store it		Verification of Hash		Distance Computation	
			Cost	Time	Cost	Time	Cost	Time	Cost	Time
BMIAE	CASIA-V3-Interval	1280	673389 ($0.3326)	1 sec	87741 gas ($0.4656)	1.5 secs	797410 gas ($0.4230)	2 secs	2751053 gas ($1.4594)	6 secs
	IITD	2560	673593 gas ($0.3573)	1 sec	1753485 gas ($0.9302)	5 secs	1673154 gas ($0.8876)	4 secs	5885037 gas ($3.1220)	10 secs
	SDUMLA-HMT	2560	673593 gas ($0.3573)	1 sec	1753485 gas ($0.9302)	5 secs	1673154 gas ($0.8876)	4 secs	5885037 gas ($3.1220)	10 secs
SviaB	-	64	412072 gas ($0.2956)	1 sec	81049 gas ($0.0581)	1 sec	40934 gas ($0.0294)	<1 sec	82956 gas ($0.0595)	<1 sec
		128	412136 gas ($0.2957)	1 sec	84218 gas ($0.0604)	1.5 secs	59081 gas ($0.0424)	1 sec	143226 gas ($0.1028)	1 sec
		256-bit	412244 gas ($0.2958)	1 sec	136915 gas ($0.0982)	2 secs	96206 gas ($0.069)	2 secs	269162 gas ($0.1931)	3 secs

Table 4.8: Computational cost in terms of number of operations

		Compute distance	CASIA-V3-Interval	IITD	SDUMLA-HMT
BMIAE	Enc/Dec	1/1	1/1	1/1	1/1
	Multiplications	$2M$-1	2559	5119	5119
	Exponentiations	0	0	0	0
SviaPA	Enc/Dec	1/1	1/1	1/1	1/1
	Multiplications	$2MN$-1	29,339	53,247	27,135
	Exponentiations	MN	14,720	26,624	13,568
SviaB	Enc/Dec	1/1	1/1	1/1	1/1
	Multiplications	$2M$-1	255	255	255
	Exponentiations	M	128	128	128

Table 4.9: Comparison of BMIAE, SviaPA & SviaB with existing approaches (EER in terms of %)

CASIA-V3-Interval	Dwivedi, R. *et al.*, [140]	0.43
	Lai, Y.L. *et al.*, [144]	0.54
	Punithavathi, P *et al.*, [141]	1.9
	Soliman, R.F *et al.*, [145]	0.63
	Zhao, D. *et al.*, [146]	1.03
	Sadhya, D. *et al.*, [148]	0.105
	BMIAE	**0.13**
	SviaPA (or) SviaB	**0.31**
IITD	Rathgeb, C., Busch, C. [156]	0.43
	Punithavathi, P *et al.*, [141]	3.3
	Gomez-Barrero, M. *et al.*, [150]	0.7
	Sadhya, D. *et al.*, [148]	1.4
	BMIAE	**0.88**
	SviaPA (or) SviaB	**0.86**
SDUMLA-HMT	Gad, R *et al.* [143]	0.300
	Kamalskar, C *et al.* [151]	2.5947
	BMIAE	**0.0002**
	SviaPA (or) SviaB	**0.13**

to other existing works. The d' comparison of BMIAE, SviaPA/SviaB with the existing approaches are shown in Table 4.10. We can infer from Table 4.10 that the genuine and imposter scores are well separated when compared to other works. The advantage of BMIAE, SviaPA and SviaB when compared to other template protection schemes is shown in Table 4.11. BMIAE, SviaPA and SviaB satisfies the properties of template protection schemes and also provides trust to the user that the cloud server/Blockchain computes the distance honestly.

Table 4.10: Comparison of BMIAE, SviaPA & SviaB with other approaches (in terms of Separability measure (d'))

	CASIA-V3-Interval	IITD
Sadhya, D. *et al.*, [148]	2.39	2.92
Walia, G.S. *et al.*, [152]	-	1.9578
BMIAE	**4.3784**	**4.3786**
SviaPA (or) SviaB	**4.3257**	**4.3877**

Table 4.11: Comparison of biometric template protection schemes with BMIAE, SviaPA & SviaB

Scheme	**Irreversibility**	**Diversity**	**Accuracy**	**Verification of Result**
Cancelable Biometrics	+	+	-	×
Biometric cryptosystems	-	-	-	×
Homomorphic Encryption	+	±	+	×
BMIAE	+	+	+	+
SviaPA	+	+	+	+
SviaB	+	+	+	+

+, -, and × indicates strongly achieved, weakly achieved and not achieved

4.5 Summary

In this chapter, three multi-instance iris authentication systems, namely BMIAE, SviaPA & SviaB are proposed to provide privacy to the iris templates and trust on the comparator result. Two different partial HE schemes, namely Paillier and ElGamal, are used to provide the privacy of the iris templates. In BMIAE & SviaB, a smart contract is used to check the similarity between encrypted reference & probe iris templates. The comparator result returned by the cloud server is verified by the public auditor in SviaPA to check whether the cloud server performs computation correctly or not. The privacy & expensive storage limitations of Blockchain for biometrics are addressed in BMIAE & SviaB. The limitations of BMIAE & SviaPA are addressed in SviaB. Experimental results prove the significance & validity of BMIAE, SviaPA & SviaB.

Chapter 5

Privacy-preserving Machine Learning based Iris Authentication on untrusted Cloud Server using FHE Scheme

The literature study about machine learning classification on encrypted data reveals that either training or classification is performed on unencrypted data leads to loss of privacy in user's data. The main contributions of this chapter are described below:

- A secure and verifiable machine learning-based iris authentication method (SvaS) is proposed. SvaS performs both privacy-preserving (PP) training & classification phases on the encrypted data. The public verifier can verify the correctness of the classification result computed by the cloud server by using a verification procedure. The nearest neighbor & multi-class perceptron classification algorithms are implemented on encrypted data and proposed two algorithms, namely private nearest neighbor (PNN) and private multi-class perceptron (PMCP).
- Proposed a feature level fusion technique, namely Contradistinguish Similarity Analysis (CSA) which increases the correlations between samples of different class and reduces the correlations between samples of the same class. It also includes a verification procedure by using polynomial factorization algorithm to verify the result returned by the cloud server.

5.1 Preliminaries

5.1.1 Classification in machine learning algorithms:

Suppose the user possess d-dimensional feature vector x, x=$(x_i)_{i=0,1,2,...,d-1,}$ where $x_i \in \mathbb{R}^d$. To classify the input x, the classification algorithm $\mathrm{C}_w(x) : \mathbb{R}^d \longmapsto C_{k^*}$ is evaluated using the model w, where $k^* \in$ [0, c) c is the number of classes. The formal definitions of two most popular classifiers namely Nearest Neighbor and Multi-class Perceptron on unencrypted data are described in the following sections.

5.1.1.1 Nearest Neighbor (NN):

NN is a non-parametric supervised classification algorithm [157, 158, 159]. During the training phase, the model stores all the training instances to make future predictions. During the classification phase, to predict the class of the probe instance, a distance measure is used between the test instance and each training instance. The most commonly used distance measures are Manhattan, Hamming, Minkowski, Chebyshev or Euclidean distance [160]. SvaS used the Manhattan distance as a distance measure which is widely used. The Model (w) selects the instance among the training instances, which is nearer to the test instance. The class label of the nearest instance will be the class label of the test instance. Suppose $X_1, X_2, \ldots, X_N$ are the N training reference templates with each X_i having d features, and Y is the probe template with d dimensions. Equation (5.1) gives the classification result of the probe template (Y).

$$C_{k^*} = argmin_{i \in [0,N)} \sum_{j=1}^{d} |X_{ij} - Y_j| \tag{5.1}$$

where $argmin$ outputs the index $i_1, i_2, .., i_N$ that makes $\sum_{j=1}^{d} |X_{ij} - Y_j|$ as small as possible.

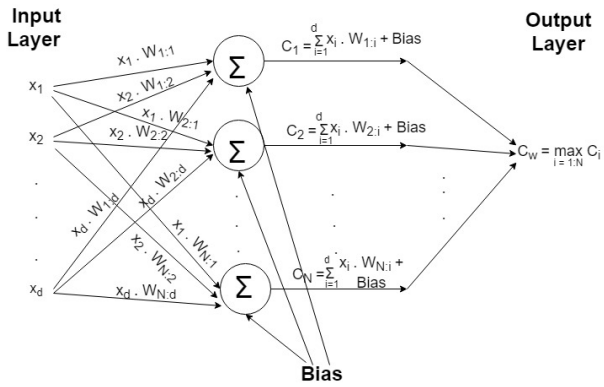


Figure 5.1: Perceptron for Multi-class Classification. $w_{i:1}, w_{i:2}, ..., w_{i:d}$ is the weight vector for the i^{th} class and $x_1, x_2, ..., x_d$ is the feature vector.

5.1.1.2 Multi-class Perceptron (MCP):

The MCP classifier [161, 162] is based on the biological model of a neuron, and it's activation value. The feature vector is multiplied (dot product) with many weight vectors. Each weight vector belongs to a class. The class label of a weight vector which yields the highest product value is the class label of the test instance. The equation to find the dot product between a feature vector and weight vector is given in the equation (5.2), where x_j, $W_{i:j}$ are the feature vector, i^{th} weight vector, $j \in [1,\ d]$ and $Bias$ is a constant which helps the model in a way that it can fit best for the given data.

$$C_i = \sum_{j=1}^{d} x_j.W_{i:j} + Bias \tag{5.2}$$

The classification result of the test instance in MCP is given in the equation (5.3). The example of MCP is shown in Figure. 5.1.

$$C_w = argmax_{i \in [0,N)} C_i \tag{5.3}$$

where $argmax$ function produces the value of i with the highest C_i value as output.

5.2 SvaS: Secure and Verifiable Machine Learning based Iris Authentication System

SvaS uses the machine learning classification to authenticate a person. The block diagram of SvaS is shown in Figure. 5.2. SvaS involves four entities, namely authentication server, cloud server, client device and public verifier. The role of authentication server is to 1) Generate secret (S_k) and public (P_k) keys. 2) Send accept/reject decision to the client device. The cloud server provides the classification service and storage to the client device. The cloud server builds a private machine learning model in the training phase and classifies the end-user using the generated model in the testing phase. The false accept/reject may happen if the cloud server doesn't perform the computations honestly. So, the correctness of the classification result computed by the cloud server is verified by the public verifier to avoid false acceptances/rejections. SvaS consists of enrollment and authentication phases. The steps involved in these phases are illustrated in Algorithm 5.1 & Algorithm 5.2.

Assumptions of SvaS

SvaS assume the following

- The client device is a trusted entity and has limited memory and computational resources.
- The authentication server is a trusted entity and generates the public, secret keys. The public and secret keys are different for each user. It broadcasts the public keys to the system, and the secret keys of the users are stored securely.
- The cloud server doesn't perform the computations honestly.
- The public verifier is only trusted to check the correctness of $\varepsilon(R)$.

5.2.1 Generation of Iris Code

This section consists of two phases, namely compression of iris template and encoding scheme. The iris template is first compressed and then encoded using the batching scheme

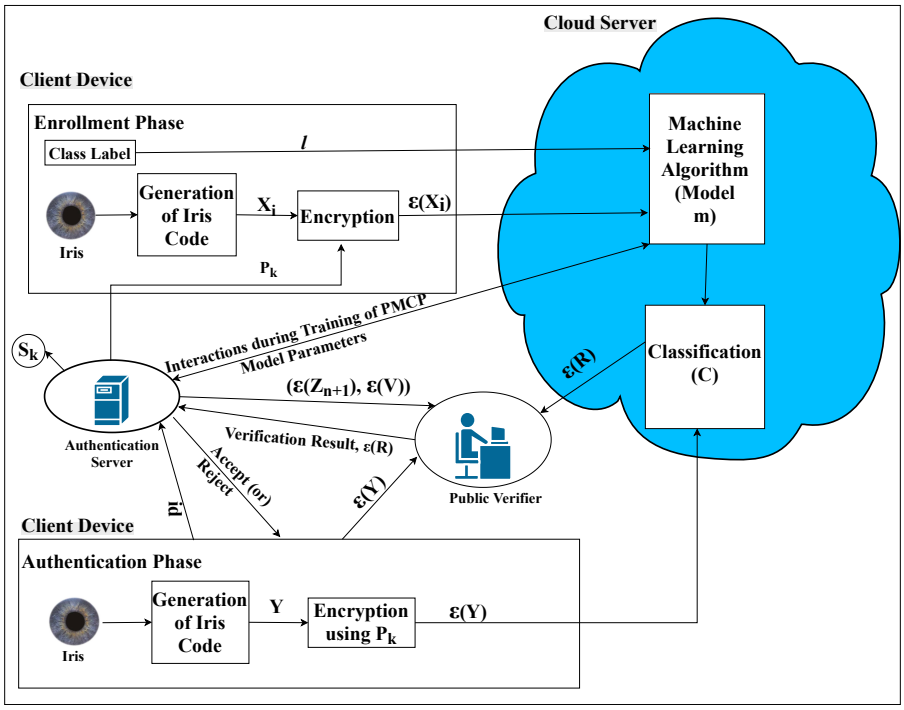


Figure 5.2: Block diagram of Secure and Verifiable Machine Learning based Iris Authentication System (SvaS)

to improve the performance of the system in terms of computational time. The size of the iris code is reduced by using the compression technique discussed in section 4.1.1.2. So, the 10240-bit binary vector is grouped into blocks of size v by using equation (4.2). v denotes the size of the block, and SvaS consider 4, 8, and 16 as v values. We can infer from Figure. 5.3 that there is a slight variation of accuracy between the actual iris code of size 1×10240 and compressed iris code of size 1×2560 for both MCP and NN. The computational time is less for compressed iris code when compared to original iris code. Hence, SvaS considers the 2560-bit iris template as a feature vector instead of the original iris code for further operations.

The compressed iris templates are encoded using the batching scheme described in section 3.1.3. The encoded polynomial is encrypted using the BFV scheme [72] described in section 3.2.1.2 to ensure the confidentiality of iris templates.

Algorithm 5.1 Enrollment Phase of SvaS

Input: Reference iris image of i^{th} user, Corresponding class label l

1: Client device generates the iris template from the reference iris image using University of Salzburg tool kit [58].
2: Client device generates the compressed iris template, X_i as described in section 5.2.1 and encode the reduced iris template as described in section 3.1.3.
3: Authentication server generate the public key, P_k and secret key S_k. *//Refer section 3.2.1.1*
4: Client device encrypts the encoded iris template and sends the encrypted reference iris template, $\varepsilon(X_i)$ along with a class label to server. *//Refer section 3.2.1.2*
5: The cloud server applies PP training on encrypted reference iris templates $\varepsilon(X_i)$, $i \in [1, N]$ using PMCP or PNN and generates a model. *//Refer section 5.2.2.1 for PNN & section 5.2.2.3 for PMCP*
6: The cloud server sends the model parameters of PMCP, $\varepsilon(w[i])$, $i \in [1, c]$ i.e., weight vectors and parameters of PNN, $\varepsilon(X_i)$, $i \in [1, N]$ to authentication server.
7: The authentication server generates the encrypted verification vector, $\varepsilon(Z_{n+1})$, encrypted random vector, $\varepsilon(V)$ separately for each classifier using the model parameters. *//Refer section 5.2.2.2 & section 5.2.2.4*

5.2.2 Secure and Verifiable Machine Learning Classification

Two private machine learning classification algorithms, namely private nearest neighbor (PNN) & private multi-class perceptron (PMCP) are implemented on encrypted data by using the homomorphic properties of BFV [72] FHE scheme. The advantage of PNN & PMCP classifiers is that they provide privacy not only to iris templates but also to the model by implementing both training & classification on the encrypted data. The model is only accessible to the server and the templates are known only to the client device.

5.2.2.1 Private Nearest Neighbor

The NN algorithm for the multi-class classification on unencrypted data is described in section 5.1. Instead of returning the class label, PNN returns the Manhattan distances between $\varepsilon(X_i)$ and $\varepsilon(Y)$.

$$R = \{r_i / r_i = \sum_{j=1}^{M} (X_i[j] - Y[j]), \forall\, i = 1\ to\ N\} \tag{5.4}$$

Algorithm 5.2 Authentication Phase of SvaS

Input: Probe iris image, Identifier or class label id of the end user
Output: Accept or Reject

1: Authentication server sends $\varepsilon(Z_{n+1})$ and $\varepsilon(V)$ to the public verifier.
2: Client device generates the iris template from probe iris image using University of Salzburg tool kit [58]. It also acquires the identifier id of the end-user and sends id to the authentication server.
3: Client device generates the compressed iris template, Y as described in section 5.2.1 and encode the reduced iris template as described in section 3.1.3.
4: Client device encrypts Y and sends the encrypted probe iris template, $\varepsilon(Y)$ to the cloud server. // *Refer section 3.2.1.2*
5: The cloud server compute the classification result, $\varepsilon(R)$ and send to public verifier. (In stead of returning the class label, our private classifiers returns the encrypted Manhattan distance between $\varepsilon(Y)$ and $\varepsilon(X_i)$ for PNN and dot product results between $\varepsilon(Y)$ and $\varepsilon(w[i])$, $i \in [1, c]$ for PMCP). // *Refer section 5.2.2.1 for PNN & section 5.2.2.3 for PMCP*
6: The public verifier checks the correctness of the computed result $\varepsilon(R)$ by using $\varepsilon(Z_{n+1})$, $\varepsilon(V)$, $\varepsilon(Y)$ and sends the verification result to authentication server. // *Refer section 5.2.2.2 & section 5.2.2.4*
7: If the verification succeeds, then the authentication server computes the predicted class label and compares with id given by the end-user to determine whether the user is genuine or not.

The server doesn't learn either $\varepsilon(X_i)$ or $\varepsilon(Y)$. In particular, we show how the server can execute equation (5.4) when both the testing & training instances are encrypted. The detailed procedure to find the NN on the encrypted data is given in Algorithm 5.3. The inputs to the PNN are the class labels of the templates, encrypted reference templates, and encrypted probe template, respectively. PNN returns the Manhattan distances, $\varepsilon(R)$ between $\varepsilon(X_i)$ and $\varepsilon(Y)$ as an output which is given in equation (5.5).

$$\varepsilon(R) = \{r_i / r_i = (\varepsilon(X_i) - \varepsilon(Y)), \forall\, i = 1\ to\ N\} \tag{5.5}$$

Since both the reference and the probe templates are in encrypted form, the privacy of iris templates, i.e., user privacy is maintained.

Let $\varepsilon(X_i)$ and $\varepsilon(Y)$ are the encrypted vectors. The aim is to achieve equation (5.5) i.e., find the Manhattan distances between $\varepsilon(X_i)$ and $\varepsilon(Y)$ without decryption. r_i is the variable to store the subtracted result of i^{th} encrypted reference template, $\varepsilon(X_i)$ and probe

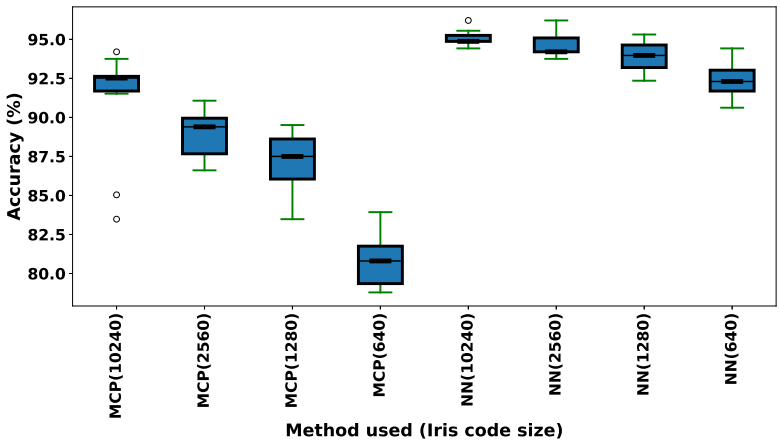


Figure 5.3: Comparison of accuracy between iris code of sizes 10240, 2560, 1280 and 640 for MCP and NN

template, $\varepsilon(Y)$. Batching scheme is used as the encoding scheme before encrypting the reference and probe templates to improve the performance of the system. Hence, with the computational cost of just one operation, we can accomplish M homomorphic subtractions. The disadvantage of batching is that it is not possible to access the individual elements of the encrypted vector. Hence, it restricts to compute the sum of elements after the subtract operation (equation (5.4)). This problem can be solved by using the observation made by Gentry *et al.* [163], particularly, it is likely to rotate the encrypted vectors cyclically without decryption. As a result, if the encrypted vectors are rotated cyclically and adding the encrypted vectors $p = log_w^q$ times then the first slot of the resultant vector gives the sum value. The steps (5-7) of Algorithm 5.3 describes the process of cyclically rotating and adding the r_i. The operation is illustrated in Figure. 5.4 with an example. The i^{th} Manhattan result is stored in r_i. The steps (3-7) of Algorithm 5.3 repeat for N reference templates yields N Manhattan distances which are assigned to $\varepsilon(R)$. The cloud server computes the Manhattan distance on the encrypted data. So, the privacy of the iris templates is achieved. If the cloud server did not perform the Manhattan distance honestly and return a random result to minimize the use of its computational resources, then false accept/reject may happen. To overcome this limitation, the public verifier checks the correctness of the result returned by the cloud server.

Algorithm 5.3 Nearest Neighbor on Encrypted data (PNN)

Input: $\varepsilon(X_1), \varepsilon(X_2), \ldots, \varepsilon(X_N)$, Corresponding class labels $cls_1, cls_2, \ldots, cls_N$, $\varepsilon(Y)$

Output: $\varepsilon(R)$

begin
  for $i \leftarrow 1$ to N **do**
    $r_i \leftarrow sub(\varepsilon(X_i), \varepsilon(Y))$
    for $j \leftarrow 0$ to p **do** *// where p = log_w^q*
      $r_i \leftarrow r_i + k_{gj}(r_i)$
    end for *//The element in the first slot is the desired Manhattan distance result*
    $r_i \leftarrow r_i$
  end for
  $\varepsilon(R) = (r_1, r_2, ..., r_N)$
return $\varepsilon(R)$
end

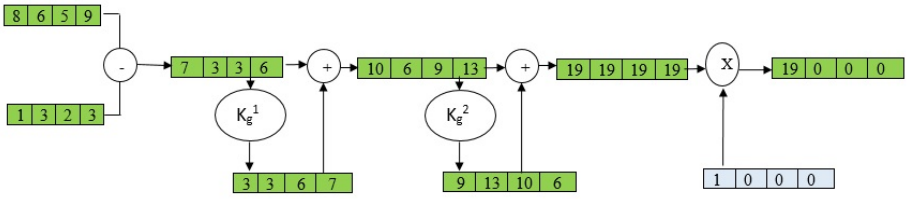


Figure 5.4: Homomorphic computation of Manhattan distance between vectors when vectors are encoded using batching scheme

5.2.2.2 Verification Scheme for Nearest Neighbor

The cloud server computes the Manhattan distances $\varepsilon(R) = r_i, \forall\, i = 1\ to\ N$ between $\varepsilon(X_i)\ \forall\, i = 1\ to\ N$ and $\varepsilon(Y)$. The verification scheme allows the public verifier to check the correctness of $\varepsilon(R)$ returned by the cloud server.

Generation of encrypted verification vector: After the enrollment phase, the authentication server constructs the encrypted verification vector using the model parameters returned by the cloud server. In the case of PNN, model parameters are simply the encrypted reference iris templates, $\varepsilon(X_i), \forall\, i = 1\ to\ N$. The encrypted verification vector helps the public verifier to check the correctness of the Manhattan distances. Let $\varepsilon(Z_{n+1})$ be the encrypted

Algorithm 5.4 Nearest Neighbor_Verification Vector

Input: $\varepsilon(X_1), \varepsilon(X_2), \ldots, \varepsilon(X_N)$,
Output: $\varepsilon(Z_{n+1}), \varepsilon(V)$

begin
  $Z_{n+1} = (0, 0, ..., 0)$
  $\varepsilon(Z_{n+1}) \leftarrow Enc(Z_{n+1}, P_k)$
  for $i \leftarrow 1$ to N **do**
    $v_i \leftarrow randomInteger()$
    $\varepsilon(v_i) \leftarrow Enc(v_i, P_k)$
    $tmp_i \leftarrow sub(\varepsilon(X_i), \varepsilon(v_i))$
    $\varepsilon(Z_{n+1}) \leftarrow add(\varepsilon(Z_{n+1}), tmp_i)$
  end for
  $\varepsilon(V) = (\varepsilon(v_1), \varepsilon(v_2), ..., \varepsilon(v_N))$
return $(\varepsilon(Z_{n+1}), \varepsilon(V))$
end

verification vector and is defined as

$$\begin{aligned} \varepsilon(Z_{n+1}) &= (\varepsilon(X_1) - \varepsilon(v_1)) + (\varepsilon(X_2) - \varepsilon(v_2)) + ... + (\varepsilon(X_N) - \varepsilon(v_N)) \\ &= \sum_{i=1}^{N} (\varepsilon(X_i) - \varepsilon(v_i)) \end{aligned} \tag{5.6}$$

where, $v_i \; \forall \; i = 1 \; to \; N$ is the random integer and $\varepsilon(V) = (\varepsilon(v_1), \varepsilon(v_2), ..., \varepsilon(v_N))$. As long as the secret key is secure, encrypted verification vector is also secure and its security relies on the hardness of RLWE described in section 3.3.2. Algorithm 5.4 explains the steps involved in the generation of encrypted verification vector for PNN.

The authentication server implements the Algorithm 5.4 after the training phase. The verification vector denoted as Z_{n+1} with same dimension of X_i is initialized to (0, 0, ..., 0). Encrypt Z_{n+1} using P_k. The function *randomInteger()* generates a random integer v_i. Encrypt v_i using P_k. The random integer generated in each and every iteration is encrypted with different public keys. The function *sub* is called to perform the subtraction between $\varepsilon(X_i)$ and $\varepsilon(v_i)$. tmp_i stores the subtraction result. The function *add* is called to perform the addition between Z_{n+1} and tmp_i. After the completion of N iterations, the encrypted verification vector $\varepsilon(Z_{n+1})$ which is shown in equation (5.6) is obtained. The N random integers are assigned to $\varepsilon(V)$. During the authentication phase, the authentication server send $\varepsilon(Z_{n+1})$ and $\varepsilon(V)$ to the public verifier.

Ensuring the correctness of Manhattan distance: The public verifier checks the correctness of Manhattan distances, $\varepsilon(R)$ using $\varepsilon(Z_{n+1})$, $\varepsilon(Y)$ and $\varepsilon(V)$. The verification scheme checks the correctness of the result on the encrypted data itself; as a result, anyone can perform the correctness of $\varepsilon(R)$ without the secret key. The steps involved to check the correctness of Manhattan distances are described in Algorithm 5.5.

Algorithm 5.5 Nearest Neighbor_Correctness

Input: $\varepsilon(Z_{n+1}), \varepsilon(V), \varepsilon(Y), \varepsilon(R)$
Output: Zero (or) Non zero

1: **begin**
2: $D2 = 0$
3: $\varepsilon(D2) \leftarrow Enc(D2, P_k)$
4: $tmp \leftarrow multiply(N, \varepsilon(Y))$
5: $D1 = sub(\varepsilon(Z_{n+1}), tmp)$
6: $\varepsilon(D1) \leftarrow Enc(D1, P_k)$
7: **for** $i \leftarrow 1$ to N **do**
8: $tmp1 \leftarrow sub(r_i, \varepsilon(v_i)), r_i \in \varepsilon(R), \varepsilon(v_i) \in \varepsilon(V)$
9: $\varepsilon(D2) \leftarrow add(\varepsilon(D2), tmp1)$
10: **end for**
11: **return** $sub(\varepsilon(D1), \varepsilon(D2))$
12: **end**

The steps (4-5) of Algorithm 5.5 computes $\varepsilon(D1) = (\varepsilon(Z_{n+1}) - N\varepsilon(Y))$. The steps (7-11) of Algorithm 5.5 computes $\varepsilon(D2) = \sum_{i=1}^{N}(r_i - \varepsilon(v_i))$. Finally, compute $(\varepsilon(D1) - \varepsilon(D2))$. If the result is zero, the Manhattan distances $\varepsilon(R)$ returned by the cloud server is considered to be correct. The below proof uses the equation (5.5), equation (5.6) and some algebraic properties of vectors and explains how $\varepsilon(D1)$ and $\varepsilon(D2)$ are same.

$$
\begin{aligned}
\varepsilon(D1) &= (\varepsilon(Z_{n+1}) - N\varepsilon(Y)) \\
&= \sum_{i=1}^{N}(\varepsilon(X_i) - \varepsilon(v_i)) - N\varepsilon(Y) \\
&= \sum_{i=1}^{N}\varepsilon(X_i) - \sum_{i=1}^{N}\varepsilon(v_i) - N\varepsilon(Y) \\
&= \sum_{i=1}^{N}\varepsilon(X_i) - N\varepsilon(Y) - \sum_{i=1}^{N}\varepsilon(v_i) \\
&= \sum_{i=1}^{N}\varepsilon(X_i) - \sum_{i=1}^{N}\varepsilon(Y) - \sum_{i=1}^{N}\varepsilon(v_i) \\
&= \sum_{i=1}^{N}(\varepsilon(X_i) - \varepsilon(Y)) - N\varepsilon(v_i) \\
&= \sum_{i=1}^{N} r_i - \sum_{i=1}^{N}\varepsilon(v_i) \\
&= \sum_{i=1}^{N}(r_i - \varepsilon(v_i)) = \varepsilon(D2)
\end{aligned}
$$

If the verification succeeds then the Manhattan distances $\varepsilon(R)$ are considered to be correct. So, the authentication server finds the predicted class by computing the index of the minimum value among $\varepsilon(R)$. The computed predicted class is compared with id given by the end-user to determine whether the user is genuine or not.

5.2.2.3 Private Multi-class Perceptron (PMCP)

The MCP algorithm for the multi-class classification on unencrypted data is described in section 5.1. Instead of returning the class label, PMCP:classification returns the dot products between $\varepsilon(w_i), \forall\ i = 1\ to\ c$ and $\varepsilon(Y)$.

$$\varepsilon(R) = \{r_i / r_i = \varepsilon(Y).\varepsilon(w[i]), \forall\ i = 1\ to\ c\} \tag{5.7}$$

Homomorphic Comparison Protocol: The procedure to compare two cipher text values without decryption is given in Algorithm. 5.6. Consider C_1, C_2 and C_a are the cipher texts for the plain texts m_1, m_2 and plain text modulus (a) are encrypted by using BFV scheme [72] respectively. The authentication server calculates $C_b = C_a + C_1 - C_2$ by using the homomorphic properties of BFV scheme [72]. The authentication server decrypts C_b and obtains the decryption result, b using S_k. The g^{th} bit, b_g of b is the comparison result, where $g = log_2 a + 1$ returned to the cloud server. If b_g = 0 then $m_1 < m_2$ otherwise $m_1 \geq m_2$. The $cmpsn$ protocol is secure because the protocol returns only one bit to the

Algorithm 5.6 Homomorphic Comparison (cmpsn)

Procedure $cmpsn(C_1, C_2)$
Input: Ciphertexts C_1, C_2
Output: b_g

1: **begin**
2: Compute $C_b = C_a + C_1 - C_2$
3: $b = Dec(S_k, C_b)$
4: **return** b_g *//b_g is the g^{th} bit of b, where $g = \log_2 a + 1$*
5: **end**

cloud server. Therefore, even in an attack scenario, the cloud server can only learn at most one single bit of the secret key. On the other hand, each time $cmpsn$ protocol is invoked by PMCP:training protocol, the authentication server uses a new secret key. So, there won't exist any leakage of secret keys to the cloud server.

This section describes about how cloud server can execute equation (5.2) and equation (5.3) when both the training and testing instances are encrypted. PMCP consists of two phases namely training phase (PMCP:training) and classification phase (PMCP:classification). The detailed procedure to find the weight vectors using MCP on encrypted data i.e., PMCP:training is given in Algorithm 5.7. The inputs to the PMCP:training are the class labels of iris templates, encrypted reference iris templates, iterations T (not encrypted) and Bias. PMCP:training returns the encrypted weight vectors as an output. The process during training phase is explained below. Let $\varepsilon(X_1)$, $\varepsilon(X_2)$, ..., $\varepsilon(X_N)$ are the encrypted reference templates. The PMCP requires multiple training iterations to fully learn the model.

During each iteration, the j^{th} encrypted reference template is multiplied with each unique weight vector and stores in ct. As explained in section 5.2.2.1, the problem with

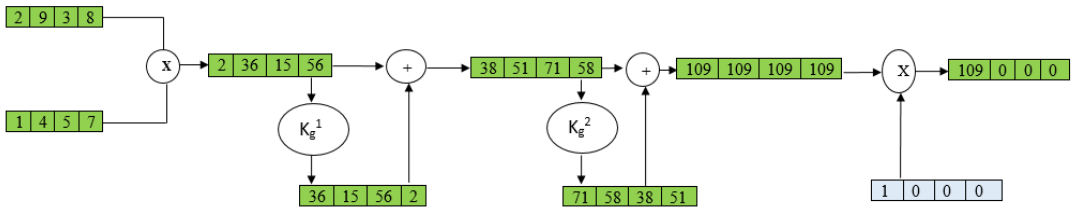


Figure 5.5: Homomorphic computation of dot product between vectors when vectors are encoded using batching scheme

batching occurs here as well while performing the sum of the elements after the multiplication. The problem can be solved by the process of cyclically rotated and adding the encrypted vectors [163]. The steps (20-22) of Algorithm 5.7 describes the process of cyclically rotating and adding the ct. The element in the first slot is the desired product. This operation is explained with an example in Figure. 5.5. The class of the j^{th} encrypted template is the class that gives the highest product result. If the calculated class, p_cls and the actual class, cls_j of the j^{th} encrypted reference template are not equal then the weight vector is updated as follows: feature vector, $\varepsilon(X_j)$ is added to the actual weight vector, $\varepsilon(w[cls_j])$ and subtracted from the predicted weight vector, $\varepsilon(w[p\_cls_j])$. After the final iteration, the final encrypted weight vectors should be stable.

The detailed procedure to achieve $\varepsilon(R)$ is given in Algorithm 5.8. The inputs to the PMCP:classification are the encrypted probe template $\varepsilon(Y)$ and encrypted weight vectors $\varepsilon(w[i]), \forall\ i = 1\ to\ c$ respectively. PMCP:classification returns the dot product $\varepsilon(R)$ between $\varepsilon(w[i]), \forall\ i = 1\ to\ c$ and $\varepsilon(Y)$ as an output which is given in equation (5.7). In Algorithm 5.8, r_i stores the multiplication result of $\varepsilon(Y)$ and $\varepsilon(w[i]), \forall\ i = 1\ to\ c$. As explained in section 5.2.2.1, the problem with batching occurs here as well while performing the sum of the elements after the multiplication. The problem can be solved by the process of cyclically rotated and adding the encrypted vectors [163]. The steps (4-6) of Algorithm 5.8 describes the process of cyclically rotating and adding the r_i. The element in the first slot is the desired dot product. The operation is explained with an example in Figure. 5.5.

The steps (3-7) of Algorithm 5.8 repeats for c times yields c dot products which are assigned to $\varepsilon(R)$. The advantage of PMCP is that the client device is not able to learn the information about the model parameters, i.e., weight vectors and the server is unable to learn

Algorithm 5.7 Perceptron for multi-class classification on encrypted data (PMCP:training)

Input: $\varepsilon(X_1)$, $\varepsilon(X_2)$, . . . , $\varepsilon(X_N)$, Corresponding class labels cls_1, cls_2, . . . ,cls_N, Iteration number T (Not Encrypted), BIAS=1 (Not Encrypted)

Output: The encrypted weight vectors for each class, $\varepsilon(w[i])$ where i ranges from 1 to c

1: **begin**
2: **for** $i \leftarrow 1$ to c **do**
3: $classes[i] \leftarrow i$
4: **end for**
5: **for** $i \leftarrow 1$ to c **do**
6: **for** $j \leftarrow 1$ to d+1 **do**
7: $w_{i,j} \leftarrow 1$
8: **end for**
9: **end for**
10: **for** $i \leftarrow 1$ to c **do**
11: $\varepsilon(w[i]) \leftarrow Enc(w_i, P_k)$ *//Batch Encryption of weight vectors*
12: **end for**
13: **for** T $iterations$ **do**
14: **for** $j \leftarrow 1$ to N **do**
15: $arg\_max \leftarrow 0$
16: $p\_cls \leftarrow classes[0]$
17: $\varepsilon(arg\_max) \leftarrow Enc(arg\_max, P_k)$
18: **for** $i \leftarrow 1$ to c **do**
19: $ct \leftarrow multiply(\varepsilon(X_j), \varepsilon(w[i]))$
20: **for** $i \leftarrow 0$ to l **do** *// where* $l = log_w^q$
21: $ct \leftarrow ct + k_{g^i}(ct)$
22: **end for** *//The element in the first slot is the desired dot product result*
23: $b \leftarrow cmpsn(ct, \varepsilon(arg\_max))$
24: **if** $b_z = 1$ **then** *//where* $z = \log_2 a + 1$,
25: $\varepsilon(arg\_max) \leftarrow ct$
26: $p\_cls \leftarrow i$
27: **end if**
28: **end for**
29: **if** $cls_j \neq p\_cls$ **then**
30: $\varepsilon(w[cls_j]) \leftarrow add(\varepsilon(w[cls_j]), \varepsilon(X_j))$
31: $\varepsilon(w[p\_cls_j]) \leftarrow sub(\varepsilon(w[p\_cls_j]), \varepsilon(X_j))$
32: **end if**
33: **end for**
34: **end for**
35: **end**

Algorithm 5.8 Perceptron for Multi-class Classification on encrypted data (PMCP:classification)

Input: $\varepsilon(Y)$, $\varepsilon(w[i])$ from training phase where i ranges from 1 to c
Output: $\varepsilon(R)$

begin
  for $i \leftarrow 1$ to c **do**
    $r_i \leftarrow multiply(\varepsilon(Y), \varepsilon(w[i]))$
    for $j \leftarrow 0$ to p **do** *// where p = log_w^q*
      $r_i \leftarrow r_i + k_{g^j}(r_i)$
    end for *//The element in the first slot is the desired dot product result*
    $r_i \leftarrow r_i$
  end for
  $\varepsilon(R) = (r_1, r_2, ..., r_c)$
return $\varepsilon(R)$
end

any information of reference templates or probe template as they are in encrypted form. Hence the privacy of both client device and model are preserved. The cloud server computes the dot products on encrypted data. So, the privacy of the iris templates is achieved. Consider a scenario; if the cloud server did not perform the dot product result honestly and return a random result to minimize the use of its computational resources, then imposter may get access into the system. To overcome this limitation, the public verifier checks the correctness of the result returned by the cloud server.

5.2.2.4 Verification Scheme for Multi-class Perceptron

The cloud server computes the dot products $\varepsilon(R) = r_i, \forall\, i = 1\ to\ c$ between encrypted weight vectors $\varepsilon(w[i]) \forall\ i = 1\ to\ c$ and $\varepsilon(Y)$. The verification scheme allows the public verifier to verify the correctness of $\varepsilon(R)$ computed by the cloud server.

Generation of encrypted verification vector: After the enrollment phase, the authentication server constructs the encrypted verification vector using the model parameters computed by the cloud server. In PMCP, model parameters are weight vectors $\varepsilon(w[i]), \forall\ i = 1\ to\ c$. The encrypted verification vector helps the public verifier to check the correctness of the dot product results.

Let $\varepsilon(Z_{n+1})$ be the encrypted verification vector and is defined as

Algorithm 5.9 Multi-class Perceptron_Verification Vector

Input: $\varepsilon(w[1]), \varepsilon(w[2]), \ldots, \varepsilon(w[c])$,
Output: $\varepsilon(Z_{n+1})$, encrypted random integers, $\varepsilon(V)$

begin
  $Z_{n+1} = (0, 0, ..., 0)$
  $\varepsilon(Z_{n+1}) \leftarrow Enc(Z_{n+1}, P_k)$
  for $i \leftarrow 1$ to c **do**
    $v_i \leftarrow randomInteger()$
    $\varepsilon(v_i) \leftarrow Enc(v_i, P_k)$
    $tmp_i \leftarrow multiply(\varepsilon(w[i]), \varepsilon(v_i))$
    $\varepsilon(Z_{n+1}) \leftarrow add(\varepsilon(Z_{n+1}), tmp_i)$
  end for
  $\varepsilon(V) = (\varepsilon(v_1), \varepsilon(v_2), ..., \varepsilon(v_c))$
return $(\varepsilon(Z_{n+1}), \varepsilon(V))$
end

$$\begin{aligned} \varepsilon(Z_{n+1}) &= \varepsilon(w[1]).\varepsilon(v_1) + \varepsilon(w[2]).\varepsilon(v_2) + ... + \varepsilon(w[c]).\varepsilon(v_c) \\ &= \sum_{i=1}^{c} (\varepsilon(w[i]).\varepsilon(v_i)) \end{aligned} \tag{5.8}$$

where, $v_i \; \forall \; i = 1 \; to \; c$ are the random integers and $\varepsilon(V) = (\varepsilon(v_1), \varepsilon(v_2, ..., \varepsilon(v_c))$. As long as the secret key is secure, encrypted verification vector is also secure and its security relies on the hardness of RLWE. The steps involved in the generation of encrypted verification vector for PMCP are given in Algorithm 5.9. The authentication server implements the Algorithm 5.9 after the training phase. The verification vector denoted as Z_{n+1} with same dimension of X_i is initialized to (0, 0, ..., 0). Encrypt Z_{n+1} using the public key P_k. The function *randomInteger()* generates a random integer which is assigned to v_i. Encrypt v_i using the public key P_k. The random integer generated in each and every iteration is encrypted with different public keys. The function *multiply* is called to perform the multiplication between encrypted weight vector $\varepsilon(w[i])$ and encrypted random integer $\varepsilon(v_i)$. tmp_i stores the multiplication result. The function *add* is called to perform the addition between Z_{n+1} and tmp_i. After the completion of c iterations, the encrypted verification vector $\varepsilon(Z_{n+1})$ which is shown in equation (5.8) is obtained. The c random integers are assigned to $\varepsilon(V)$. During the authentication phase, the authentication server send $\varepsilon(Z_{n+1})$

Algorithm 5.10 Multi-class Perceptron_Correctness

Input: $\varepsilon(Z_{n+1}), \varepsilon(V), \varepsilon(Y), \varepsilon(R)$
Output: Zero (or) Non zero

1: **begin**
2: $D2 \leftarrow 0$
3: $\varepsilon(D2) \leftarrow Enc(D2, P_k)$
4: $\varepsilon(D1) = multiply(\varepsilon(Z_{n+1}), \varepsilon(Y))$
5: **for** $j \leftarrow 0$ to p **do** *// where p = log_w^q*
6: $\varepsilon(D1) \leftarrow \varepsilon(D1) + k_{gj}(\varepsilon(D1))$
7: **end for** *//The element in the first slot is the desired result*
8: **for** $i \leftarrow 1$ to c **do**
9: $tmp1 \leftarrow multiply(r_i, \varepsilon(v_i)), r_i \in \varepsilon(R), \varepsilon(v_i) \in \varepsilon(V)$
10: $\varepsilon(D2) \leftarrow add(\varepsilon(D2), tmp1)$
11: **end for**
12: **return** $sub(\varepsilon(D1), \varepsilon(D2))$
13: **end**

and $\varepsilon(V)$ to the public verifier.

Ensuring the correctness of dot product: The public verifier checks the correctness of dot products $\varepsilon(R)$ using $\varepsilon(Z_{n+1})$, $\varepsilon(Y)$ and $\varepsilon(V)$. Our verification scheme checks the correctness of the result on the encrypted data itself as a result anyone can perform the correctness of the $\varepsilon(R)$ without the private information. The steps involved to check the correctness of dot products are described in Algorithm 5.10.

The steps (4-7) of Algorithm 5.10 computes $\varepsilon(D1) = \varepsilon(Z_{n+1}).\varepsilon(Y)$. The steps (8-11) of Algorithm 5.10 computes $\varepsilon(D2) = \sum_{i=1}^{c}(r_i.\varepsilon(v_i))$. Finally, compute $(\varepsilon(D1) - \varepsilon(D2)$. If the result is zero, then the dot product values $\varepsilon(R)$ returned by the cloud server is considered to be correct. Equation (5.9) uses the equation (5.7), equation (5.8) and some algebraic properties of vectors and explains how $\varepsilon(D1)$ and $\varepsilon(D2)$ are same. If the verification succeeds then the dot products $\varepsilon(R)$ are correct. So, the authentication server computes the predicted class by computing the index of the maximum value among $\varepsilon(R)$. The computed predicted class is compared with id given by the end user to determine whether the user is genuine or not.

$$\begin{aligned}
\varepsilon(D1) &= \varepsilon(Y).\varepsilon(Z_{n+1}) \\
&= \varepsilon(Y).\sum_{i=1}^{c}(\varepsilon(w[i]).\varepsilon(v_i)) \\
&= \sum_{i=1}^{c}(\varepsilon(w[i]).\varepsilon(Y).\varepsilon(v_i)) \\
&= \sum_{i=1}^{c}\varepsilon(v_i).(\varepsilon(w[i].\varepsilon(Y)) \\
&= \sum_{i=1}^{c}\varepsilon(v_i).r_i \\
&= \varepsilon(D2)
\end{aligned} \tag{5.9}$$

5.3 Multi-instance Iris Remote Authentication using private multi-class perceptron on Malicious Cloud Server (MIRAMCS)

MIRAMCS uses the multi-class perceptron classification to authenticate a person. The block diagram of MIRAMCS is shown in Figure. 5.6. MIRAMCS involves four entities, namely authentication server, client device, cloud server and public verifier. The role of authentication server is to 1) Generate secret (S_k) and public (P_k) keys. 2) Send accept/reject decision to the client device. The classification service and storage to the client device is provided by the cloud server. During the training phase, the cloud server builds a private machine learning model & classifies the end-user using the developed model in the classification phase. The false accept/reject may happen if the cloud server doesn't perform the computations honestly. So, the correctness of the classification result computed by the cloud server is verified by the public verifier to avoid false acceptances/rejections. MIRAMCS consists of enrollment and authentication phases. The steps involved in these phases are illustrated in Figure. 5.7 and Figure. 5.8.

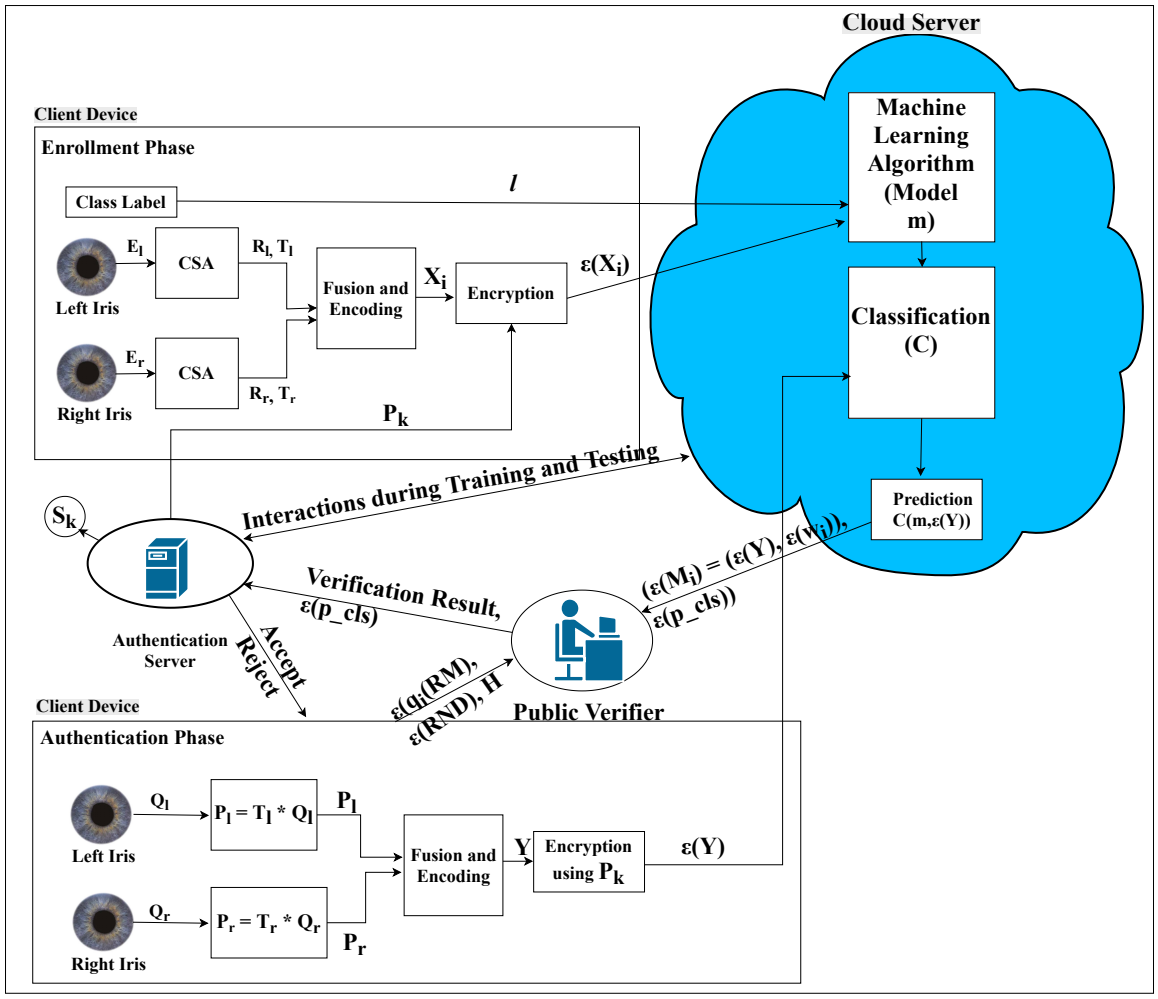


Figure 5.6: Block diagram of Multi-instance Iris Remote Authentication using private multi-class perceptron on Malicious Cloud Server (MIRAMCS)

Assumptions of MIRAMCS

MIRAMCS assume the following

- The client device is a trusted entity and has limited memory and computational resources.
- The authentication server is a trusted entity and generates the public, secret keys. The public and secret keys are different for each user. It broadcasts the public keys to the system, and the secret keys of the users are stored securely.
- The cloud server doesn't perform the computations honestly.
- The public verifier is only trusted to check the correctness of $\varepsilon(R)$.

Enrollment Phase		
Authentication Server	**Client Device**	**Malicious Cloud Server**
Key Generation (P_k, S_k)	1. Acquire the reference left and right iris images of s separate users and generate corresponding iris codes E_l, E_r. 2. Apply CSA on the samples in the data matrix which are collected from s separate classes to obtain transformation matrices T_l, T_r and transformed iris codes R_l, R_r. 3. Apply fusion and encoding on R_l, R_r to get the fused reference template X_i which contains discriminative information. 4. $\varepsilon(X_i)$ = Enc(P_k, X_i) and send $\varepsilon(X_i)$ to the cloud server	
		Store $\varepsilon(X_i)$ in database. Apply the PP training on $\varepsilon(X_i)$ using PMCP and generates a secure model.

Figure 5.7: Enrollment Phase of Multi-instance Iris Remote Authentication using private multi-class perceptron on Malicious Cloud Server (MIRAMCS)

5.3.1 Contradistinguish Similarity Analysis (CSA)

CSA maximizes the pair-wise correlations & minimizes the between-class correlations. CSA also includes the class structure similar to discriminant correlation analysis (DCA). The difference between DCA and CSA is in the way of defining the between-class scatter matrix and covariance matrix. Assume the data matrix consists of N samples belong to c different classes. The feature vector of k^{th} sample in l^{th} class is represented by a_k^l. The nearest neighbor to a_k^l belongs to k^{th} class & not belongs to k^{th} class denoted by $(a_k^l)_w$ & $(a_k^l)_b$. The weight vector corresponding to (a_k^l) is denoted as $v(k,\ l)$ and is defined in

Authentication Phase			
Client Device	**Authentication Server**	**Public Verifier**	**Malicious Cloud Server**
1. Acquire the probe left and right iris image and generate corresponding iris codes Q_l, Q_r.			
2. Multiply the transformation matrices T_l, T_r with Q_l, Q_r to obtain transformed iris code P_l, P_r			
3. Apply fusion and encoding on P_l, P_r to get the fused probe template Y which contains discriminative information.			
4. Encrypt the probe template. $\varepsilon(Y)$ = Enc(P_k, Y) and send $\varepsilon(Y)$ to the cloud server.			
			5. Compute the classification result, $\varepsilon(p\_cls)$ using PMCP.
			6. Send $\varepsilon(p\_cls)$, $\varepsilon(Y)$, $\varepsilon(w[c])$ to the public verifier.
7. Prepare Parameters for Verification (Explained in section 5.3.3) and send these parameters to the public verifier.			
		8. Result Verification using $\varepsilon(b_j(RM)_{j=1}^{c})$, $\varepsilon(RND)$, H, $\varepsilon(p\_cls)$, $\varepsilon(Y)$, $\varepsilon(w[c])$ (Explained in section 5.3.3) and send verfication result to the authentication server	
	9. If the verification succeeds, then decrypt the classification result to determine whether the user is genuine or not and send accept/reject to the client device		

Figure 5.8: Authentication Phase of Multi-instance Iris Remote Authentication using private multi-class perceptron on Malicious Cloud Server (MIRAMCS)

equation (5.12). The between-class scatter matrix (BCSM) is defined as

$$B_{sr} = \sum_{k=1}^{c} \sum_{l=1}^{N_k} v(k,\ l)(a_l^k - (a_l^k)_b)(a_l^k - (a_l^k)_b)^T = \delta_{sr}\delta_{sr}^T \tag{5.10}$$

where

$$\delta_{cr} = [\sqrt{v(1,1)}(a_1^1 - (a_1^1)_b, \sqrt{v(1,2)}(a_2^1 - (a_2^1)_b, ..., \sqrt{v(c,N)}(a_N^c - (a_N^c)_b] \tag{5.11}$$

and

$$v(k,l) = \frac{min\{d(a_k^l, (a_k^l)_w), d(a_k^l, (a_k^l)_b)}{d(a_k^l, (a_k^l)_w) + d(a_k^l, (a_k^l)_b)} \tag{5.12}$$

The euclidean distance between two vectors m & n is denoted as $d(m,n)$. The BCSM can be diagonalized as follows:

$$X^T B_{sr} X = \phi \tag{5.13}$$

Where X represents the right eigenvectors of B_{sr} and ϕ represents the diagonal matrix containing eigenvalues in decreasing order corresponding to the eigenvectors. The dimensions are reduced to c-1 in DCA [164] whereas in CSA, the top t eigen values & their corresponding eigen vectors are chosen to preserve the significant dimension of the feature vector for correlation analysis.

$$X_{(t\times d)}^T B_{sr} X_{(d\times t)} = \phi_{(t\times t)} \tag{5.14}$$

The size of the data matrix R reduces from d to t.

$$R'_{(t\times N)} = X_{(t\times d)}^T R_{(d\times N)} \tag{5.15}$$

Similarly, the other feature vector is solved & find the transformed feature vector that diagonalizes the BCSM B_{sp}. The P is transformed to P'.

$$P'_{(t\times N)} = X_{(t\times d)}^T P_{(d\times N)} \tag{5.16}$$

The diagonal & non-diagonal elements of ${\delta'_{sr}}^T \delta'_{sr}$ & ${\delta'_{sp}}^T \delta'_{sp}$ are nearer to one and zero. This indicates the matrices are strict diagonally dominant. Therefore, the classes are well separated because of less correlation between the centroid of the classes.

A class matrix $Z \in \mathbb{R}^{N \times c}$, where each row of Z denotes the class label. class 1, class 2, ... , class s are represented as [1 0 0 0..], [0 1 0 0..], ..., [0 0 0...1]. The covariance matrix of R' & P' are given as:

$$V_c = (R'DZ)(P'DZ)^T \tag{5.17}$$

where $D \in \mathbb{R}^{N \times N}$ & is given as $D = I - N^{-1}ii^T$, i is an identity vector. V_c is diagonalized by using singular value decomposition (SVD) to obtain non-zero correlation between corresponding features in both the feature vectors.

$$V_c = I\, A\, J^T \tag{5.18}$$

where A is a diagonal matrix of singular values
I & J contains left & right singular vectors corresponding to singular values of A.
Equation (5.18) can be rewritten as

$$I^T\, V_c\, J = A \tag{5.19}$$

Assume $L_r = I\, A^{-1/2}$ & $L_p = J\, A^{-1/2}$, then V_c can be unitized as

$$(I\, A^{-1/2})^T\, V_c (J\, A^{-1/2}) = I \Rightarrow (L_r)^T\, V_c\, (L_p) = I \tag{5.20}$$

The dimension of both L_r & L_p are $t \times t$. Since, c-1 dimensions only contribute for transformation the dimensions are reduced from t to c-1. So, the feature vectors can be transformed as follows:

$$R'' = L_r^T\, R' = L_r^T\, X_r^T\, R = W_r\, R \tag{5.21}$$

$$P'' = L_p^T\, P' = L_p^T\, X_p^T\, P = W_p\, P \tag{5.22}$$

CSA produces the transformed matrices and transformed features sets as outputs which is shown in Figure. 5.6. During the enrollment phase, CSA takes E_l, E_r as input and produces

transformed reference iris templates R_l, R_r & transformed matrices T_l, T_r. The probe iris templates Q_l, Q_r are multiplied with T_l, T_r to produce transformed probe iris templates P_l, P_r.

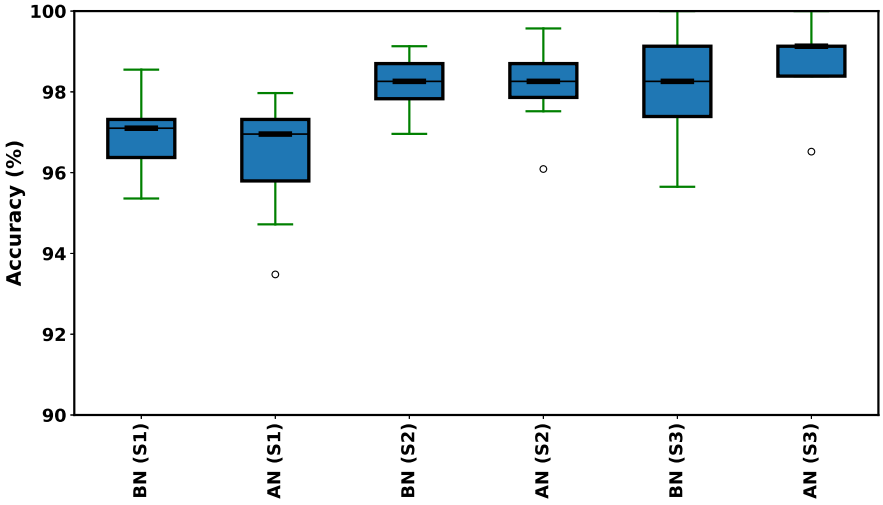
5.3.2 Fusion & Encoding

The transformed iris templates obtained in CSA are fused using the technique discussed in section 4.1.1.1. The fused iris template is encoded using the batching scheme discussed in section 3.1.3 to improve the performance of the system. The batching scheme encodes a group of integers into a single polynomial but the fused iris template contains non-integer values. So, the scaling of fused feature sets has to be done before encoding. Min-Max normalization is used for feature scaling. The comparison of accuracy between before normalization and after normalization for CASIA-V3-Interval and IITD iris database is shown in Figure. 5.9. The accuracies obtained after normalization are less when compared to accuracies obtained before normalization, but batching helps to reduce the time complexity and improves the performance of the system. So, MIRAMCS considered the values obtained after normalization for further operations.

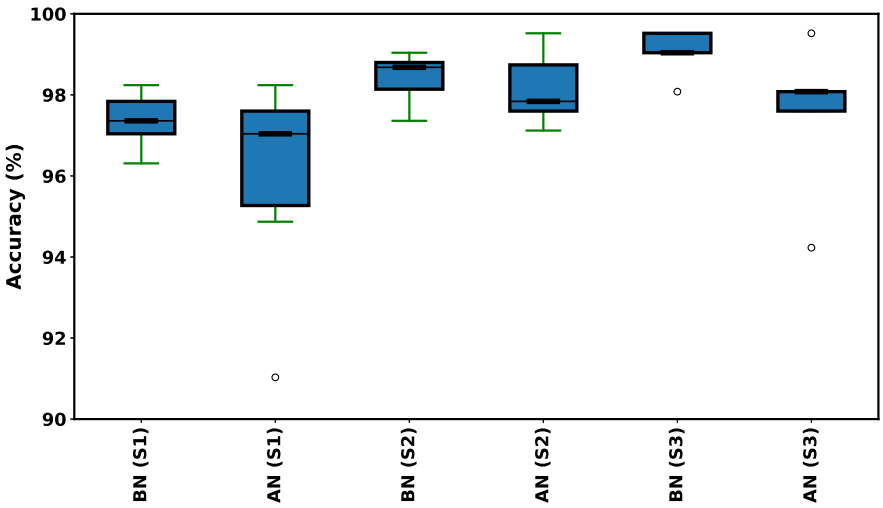
The cloud server classifies the encrypted fused probe iris template by using PMCP discussed in section 5.2.2.3. If the cloud server doesn't perform the classification honestly then false accept/reject may happen. So, a verification procedure is described in section 5.3.3, in which a public verifier checks the classification result returned by the cloud server.

5.3.3 Verification Procedure

The public verifier & the authentication server receives $\varepsilon(p_{cls})$ from the cloud server. To save the computational time & other resources, the cloud server may send arbitrary classification result without performing the desired computation results in false accept/reject. The public verifier helps to check the correctness of the classification result computed by the cloud server. The verification parameters, classification parameters & encrypted probe template are used by the public verifier to check the correctness of the classification result. The authentications server receives the verification result from the public verifier. The pre-



(a)



(b)

Figure 5.9: Comparison of before-normalization (BN) and after-normalization (AN) accuracies with different train-test split ratios (S1, S2 and S3 are described in Section 5.4.1) for a) CASIA-V3-Interval b) IITD database

dicted class, $\varepsilon(p_{cls})$ is correct if the verification succeeds. Later, $\varepsilon(p_{cls})$ is decrypted by the authentication server using P_k & send the result to the client device. The predicted class is not computed correctly by the cloud server if the verification fails.

Multivariate Polynomial Factorization (MVPF):

Consider $h(x) = h(a_1, a_2, ..., a_n) \in \mathbb{Z}_p^n[y]$ as a m-variable polynomial. According to MVPF, $h(x) - h(a)$ can be expressed as $h(x) - h(a) = \sum_{j=1}^{m}(x_j - a_j)b_j(x)\ \forall a \in \mathbb{Z}_p^n \exists b_i(x) \in \mathbb{Z}_p^n[x]$. A polynomial-time algorithm exists to find $b_i(x)$.

Construction of Verification Parameters:

Consider M_k = (a_{kl}, b_l), where $k \in$ [1, c] and $j \in$ [1, M]. The client device decomposes $H^{'}(y) = H(y) - H(N_1,\ N_2,\ ...,\ N_c)$ into $\sum_{j=1}^{c}(y_j - N_j)b_j(y)$ by using the MVPF, and $y \in \mathbb{Z}_p^c$ and $b_j(y)_{j=1}^{c}$ are the polynomials generated by the MVPF. Client device selects a set of random data at the same time, $RM_k = (RA_{kl},\ RB_l)$, $k \in$ [1, c] & $j \in$ [1, M]. This random data is refreshed periodically. The client device calculates $RND = H(RM_1,\ RM_2,\ ...,\ RM_c)\ and\ b_i(RM)_{i=1}^{c}$. The client device sends $(\varepsilon(b_j(RM)_{j=1}^{c}),\ \varepsilon(RND),\ H)$ to the public verifier.

Verification of predicted result ($\varepsilon(p\_cls)$)

The public verifier collects (($\varepsilon(D_j)$ = ($\varepsilon(w_j)$, $\varepsilon(Y)$)), $\varepsilon(p\_cls)$) from the cloud server and ($\varepsilon(b_j(RM)_{j=1}^{c})$, $\varepsilon(RND)$, H) from the client device. The main operation of the verification process is to calculate the polynomial factorization formula in a fully homomorphic manner. Our verification procedure reduces the difficulty of the user by allowing anyone can check the correctness of the classification result without the need for user's secret keys by using FHE. The public verifier checks whether equation (5.23) holds or not.

$$Eval\{\varepsilon(RND) - \varepsilon(p\_cls)\} \stackrel{?}{=} Eval\{\sum_{j=1}^{c}(\varepsilon(RM_j) - \varepsilon(D_j))\varepsilon(b_j(RM)\} \tag{5.23}$$

5.4 Implementation details and Security Analysis of SvaS, MIRAMCS

The following measures are used to evaluate the efficiency of a biometric system according to biometric information protection [23].

1. Performance evaluation in terms of EER, d' and KS-test.
2. Irreversibility and Unlinkability Analysis.
3. Computational cost in terms of time taken to perform operations.

5.4.1 Performance Evaluation of SvaS and MIRAMCS

The classification accuracy of SvaS with PNN & PMCP for different databases with different train-test split ratios are shown in Figure. 5.10. The comparison of accuracy between protected & unprotected templates of SvaS for different databases when train-test split ratio is 60-40 is shown in Figure. 5.11. From Figure. 5.11, we infer that there is no degradation of accuracy between protected & unprotected templates in SvaS.

The accuracy, training & classification time on normal data for only left iris (OLI), only right iris (ORI), fusion without CSA (FWCSA) & fusion using CSA (FUCSA) for MIRAMCS is shown in Table 5.1. The accuracies of canonical correlation analysis (CCA), DCA & CSA with different train-test ratios for CASIA-V3-Interval, IITD iris databases are shown in Figure. 5.12. From Figure. 5.12, we infer that CSA performs better than CCA & DCA. The comparison of accuracy between protected & unprotected templates of MIRAMCS for different databases for different train-test split ratios is shown in Figure. 5.13. From Figure. 5.13, we infer that there is no degradation of accuracy between protected & unprotected templates in MIRAMCS.

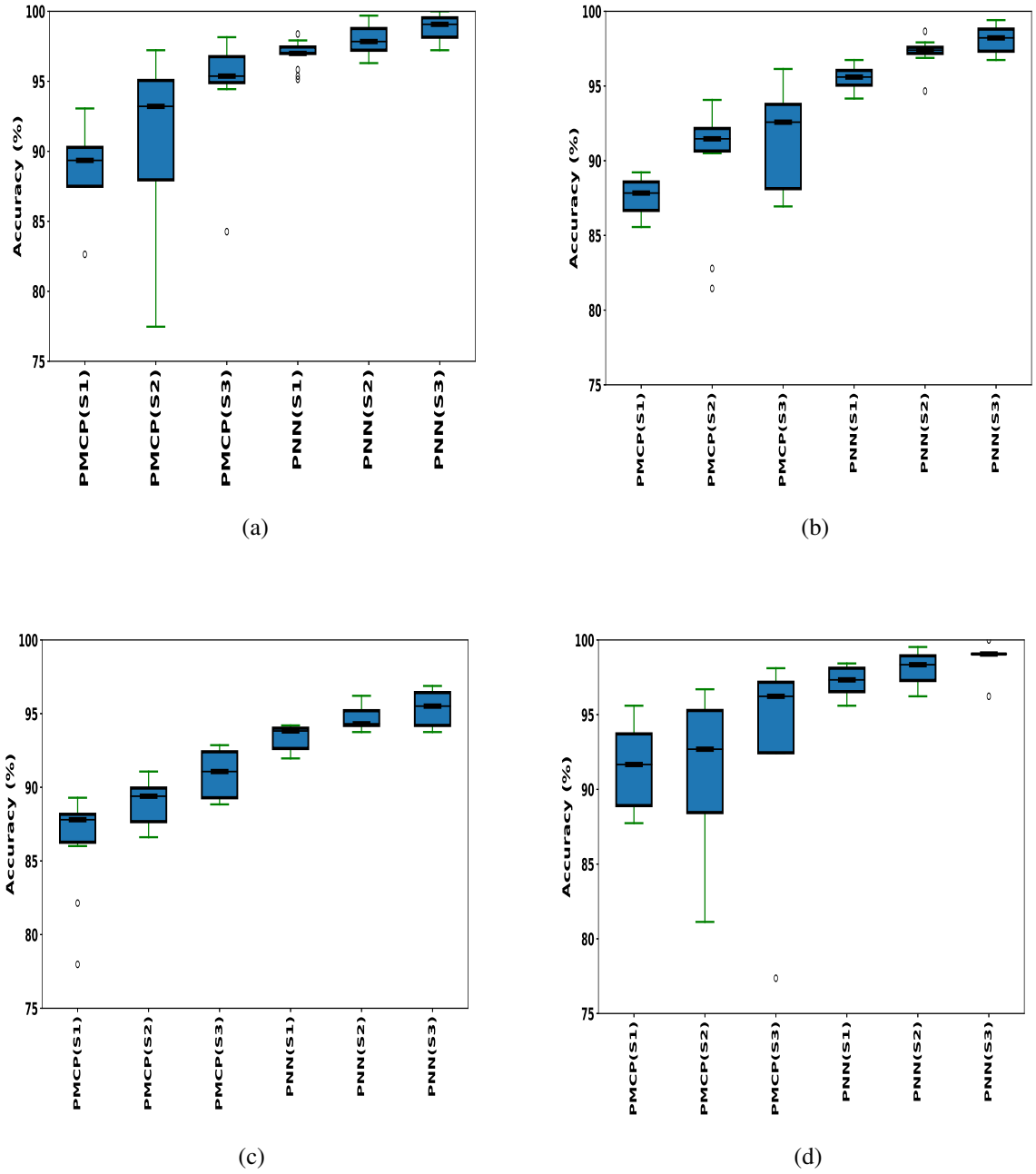
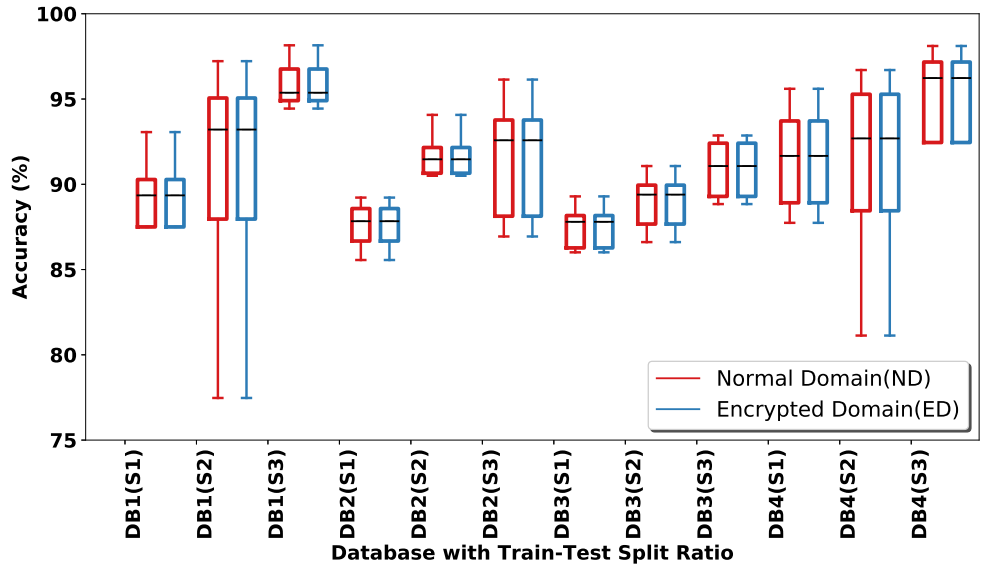
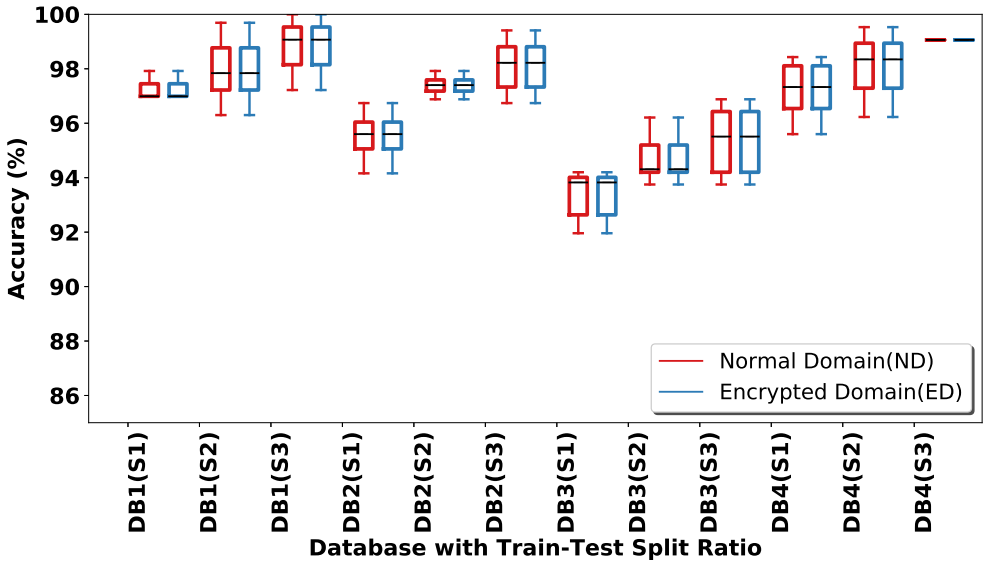


Figure 5.10: Accuracy of SvaS (PMCP & PNN with different train-test split ratio) obtained for a) CASIA-V 1.0 b) CASIA-V3-Interval c) IITD d) SDUMLA-HMT iris databases

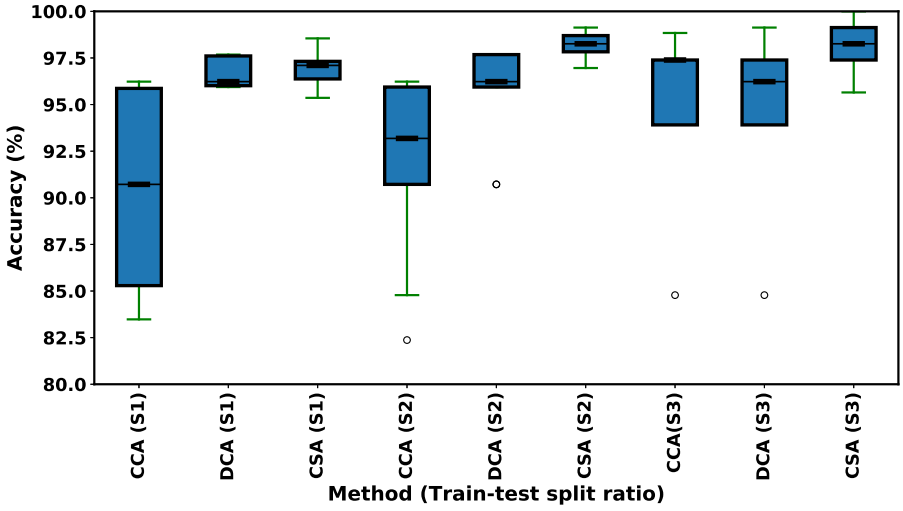


(a)

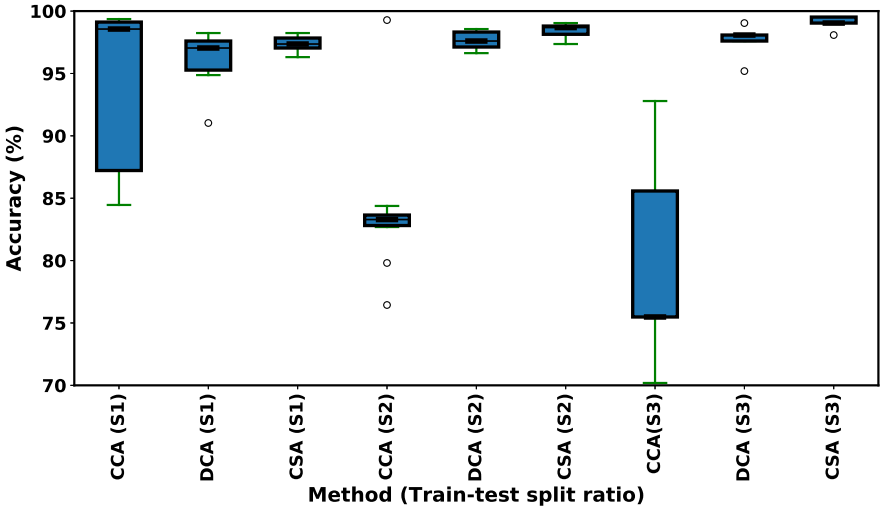


(b)

Figure 5.11: Comparison of accuracy of SvaS between protected and unprotected templates for a) MCP b) NN; **DB1**: CASIA-V 1.0, **DB2**: CASIA-V3-Interval, **DB3**: IITD & **DB4**: SDUMLA-HMT



(a)



(b)

Figure 5.12: Average classification accuracy of MIRAMCS obtained for a) CASIA-V3-Interval b) IITD iris databases

Table 5.1: Accuracy obtained in unprotected system for MIRAMCS (MCP classifer with 80-20 train-test ratio)

Instance	CASIA-V3-Interval					IITD				
	T	f	TRT (secs)	TST (secs)	Accuracy (%)	T	f	TRT (secs)	TST (secs)	Accuracy (%)
OLI	100	10240	170.66	0.005	94.09	100	10240	368.37	0.01	91.79
ORI	100	10240	170.66	0.005	93.22	100	10240	368.37	0.01	93.14
FWCSA	100	20480	325.89	0.008	96.21	100	20480	695.08	0.009	95.34
FUCSA	400	228	35.39	0.0006	98.15	400	414	79.37	0.0007	97.15

f refers to size of iris template.
TRT refers to training time (in seconds).
TST refers to testing time (in seconds).

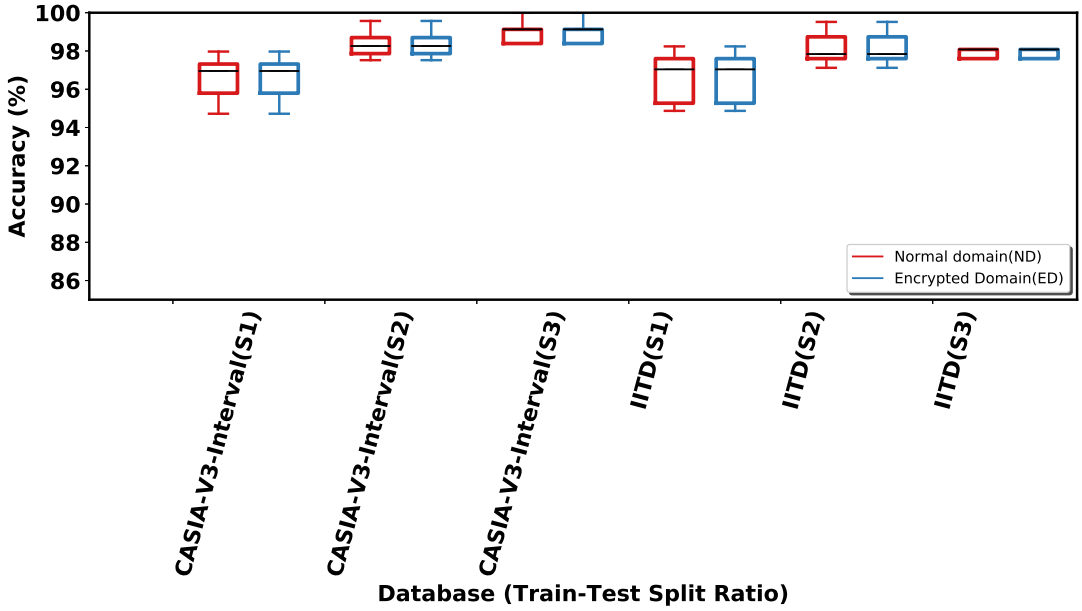


Figure 5.13: Comparison of accuracy of MIRAMCS between protected and unprotected templates

5.4.2 Security Analysis of SvaS & MIRAMCS

The template protection method must satisfy the requirements of irreversibility, revocability and unlinkability to ensure the privacy of the iris templates. The vulnerability of attacks in SvaS and MIRAMCS can occur in the following entries:

1. The cloud server.
2. The client device.
3. The communication channel between the cloud server and the client device.
4. The authentication server.
5. The public verifier.

The client device extracts the features of the iris image. Hence, security is to be ensured for the client device. As, SvaS and MIRAMCS assume the client device is a trusted entity, the features of iris image are secure. The authentication server generates the keys needed for encryption and decryption. SvaS & MIRAMCS assume that the authentication server is also a trusted entity. Since the security of SvaS & MIRAMCS depends on the apparent hardness of Ring Learning with Errors (RLWE) problem, the iris templates stored in the server database are secure. It is difficult to decrypt the encrypted iris templates without the secret key. As a result, the communication channel is also reliable. The description of RLWE is given in section 3.3.2.

Irreversibility Analysis: Irreversibility refers to obtaining the original template from the encrypted template. The client device sends the encrypted reference templates to the cloud server during the enrollment phase, and encrypted probe iris template of a user to the server for classification result. The server classifies the encrypted probe template and returns the encrypted classification result to the authentication server. As the SvaS & MIRAMCS uses BFV scheme to protect the templates, and the security of BFV scheme relies on solving the RLWE problem, it is computationally infeasible to decrypt the templates by the server or an imposter without secret key (S_k). Therefore, SvaS & MIRAMCS satisfies the irreversibility property.

Revocability Analysis: Revocability ensures that a new protected template should be generated by the protection method if the old template is compromised or stolen. In SvaS & MIRAMCS, Revocability can be achieved by re-encrypting the samples in the database with a new key pair (P'_k, S'_k) instead of acquiring the new samples from the users.

Unlinkability Analysis: Unlinkability ensures that there won't be any correlation between the protected templates used in different applications. BFV scheme used in SvaS & MIRAMCS is based on probabilistic encryption. Due to the randomness involved in BFV scheme, different ciphertexts can be generated even if the same message is encrypted multiple times with the same key, and there won't exist any similarity between the generated ciphertexts.

5.4.3 Computational Analysis of SvaS & MIRAMCS

For a given desired security level (λ), the time taken (in seconds) to encrypt, decrypt and to classify the encrypted probe template for different security parameter values and iris code sizes of SvaS is given in Table 5.2. The average time in seconds by running the experiments ten times is considered. The table also shows the time taken to perform classification on unencrypted values. From Table 5.3, we infer that the reduction in the size of the iris template and batching scheme can speed-up homomorphic iris computation over element-wise (without batching scheme). The iris template size is proportional to the computational time. SvaS converts 1×10240 into 1×640, 1×1280, 1×2560 respectively. Even though the total time taken for iris code of size 640, and 1280 is less when compared to iris code of size 2560, but the optimal accuracy is achieved with iris template of size 1×2560. For a given desired security level (λ), the time taken (in seconds) to encrypt, decrypt and to classify the encrypted fused probe template in MIRAMCS for n = 4096 & a = 40961 is given in Table 5.3.

Table 5.2: Total time taken in SvaS (with & without batching scheme)

	Security (λ)	M	**NFT**	**Parameters**			**Time with Batching scheme (sec)**				**Time without Batching scheme (sec)**			
				n	q (bits)	x	Enc	Score	Dec	Total	Enc	Score	Dec	Total
PMCP	128-bit	640	0.16	1024	29	40961	0.003	1.19	0.0008	1.1938	2.688	482.262	0.64	485.59
		1280	0.3	2048	56	40961	0.005	2.1	0.0018	2.1068	5.376	964.525	1.28	971.181
		2560	0.38	4096	110	40961	0.011	4.13	0.0038	4.1448	12.54	1929.05	4.096	1945.690
	192-bit	640	0.16	1024	20	40961	0.004	1.21	0.0009	1.2149	2.688	483.262	0.768	486.7175
		1280	0.3	2048	39	40961	0.005	2.2	0.0016	2.2066	5.504	964.825	1.664	971.993
		2560	0.38	4096	77	40961	0.013	4.15	0.0040	4.167	12.8	1930.21	4.096	1947.108
PNN	128-bit	640	0.15	1024	29	40961	0.003	2.24	0.0008	2.2438	2.688	680.393	0.64	683.721
		1280	0.27	2048	56	40961	0.005	4.4	0.0018	4.4068	5.376	1360.786	1.28	1367.442
		2560	0.52	4096	110	40961	0.011	13.3	0.0038	13.3148	12.54	2721.60	4.096	2738.243
	192-bit	640	0.15	1024	20	40961	0.004	2.12	0.0009	2.1249	2.688	681.342	0.768	684.849
		1280	0.27	2048	39	40961	0.005	4.4	0.0016	3.4666	5.504	1360.796	1.664	1367.964
		2560	0.52	4096	77	40961	0.013	13.71	0.0040	13.727	12.8	2722.60	4.096	2739.499

λ refers to security.
M refers to size of iris template.
q refers to coefficient modulus.
NFT refers to Time in seconds without FHE.
Enc, Score and Dec stands for time taken to perform encryption, prediction and decryption.

Table 5.3: Total time taken in MIRAMCS (for n = 4096 & x = 40961)

Database	λ	M	q	**NFT**	**Time with Batching scheme (sec)**				**Time without Batching scheme (sec)**			
					Enc	Score	Dec	Total	Enc	Score	Dec	Total
CASIA-V3-Interval	128-bit	228	110	0.006	0.004	2.16	0.003	2.167	0.96	182.39	0.12	183.47
	192-bit	228	77	0.006	0.005	2.25	0.004	2.26	0.97	182.73	0.27	183.97
IITD	128-bit	414	110	0.007	0.004	4.86	0.003	4.867	1.74	340.13	0.21	342.08
	192-bit	414	77	0.007	0.005	4.69	0.004	4.70	1.76	340.75	0.50	343.01

λ refers to security.
M refers to size of the iris template after CSA.
q refers to coefficient modulus.
NFT refers to Time in seconds without FHE.
Enc, Score and Dec stands for time taken to perform encryption, prediction and decryption.

Table 5.4: Comparison Analysis in terms of Accuracy for CASIA-V3-Interval & IITD database

Method	**CASIA-V3-Interval**	**IITD**
Rathgeb *et al.* [165]	-	97%
Sardar *et al.* [166]	97.12%	97.19%
Barpanda *et al.* [147]	91.65%	89.72%
Arsalan *et al.* [90]	99.10%	98.41%
Zhao *et al.* [91]	96.92%	96.80%
Noruzi *et al.* [167]	98.80%	99.57%
SvaS (MCP)	**91.52%**	**90.89%**
SvaS (NN)	**98.12%**	**97.35%**
MIRAMCS	**98.15%**	**97.95%**

Table 5.5: Comparison of biometric template protection schemes with SvaS and MIRAMCS

Scheme	**Irreversibility**	**Diversity**	**Accuracy**	**Verification of Result**
Cancelable Biometrics	+	+	-	×
Biometric cryptosystems	-	-	-	×
Homomorphic Encryption	+	±	+	×
SvaS	+	+	+	+
MIRAMCS	+	+	+	+

+, -, and × indicates strongly achieved, weakly achieved and not achieved

5.4.4 Comparison Analysis of SvaS & MIRAMCS

The accuracy comparison of SvaS and MIRAMCS with state-of-the-art works is shown in Table 5.4. We can infer that SvaS & MIRAMCS shows better performance when compared to [165, 166, 147, 91] and lesser performance when compared to [90, 167], which are devoid of guarantee the properties of BTP schemes. The accuracies obtained in SvaS with MCP and NN are mentioned in Table 5.4. We also observe from Table 5.4 that MIRAMCS performs better when compared to SvaS. The increase in the performance is due to feature level fusion technique (CSA) involved in MIRAMCS.

The advantage of SvaS & MIRAMCS when compared to other template protection schemes is shown in Table 5.5. SvaS & MIRAMCS satisfies the properties of template protection schemes and also provides trust to the user that the cloud server computes the distance honestly.

5.5 Summary

In this chapter, two iris authentication systems, namely SvaS & MIRAMCS, are proposed to provide the privacy to the iris templates and trust on the comparator result. SvaS & MIRAMCS uses the machine learning classification methods to authenticate a person. MIRAMCS is a multi-instance iris authentication system. The BFV FHE scheme is used to provide the privacy of the iris templates. Two private machine learning classification algorithms, namely private nearest neighbor & private multi-class perceptron are implemented on encrypted data by using the homomorphic properties of BFV scheme. In MIRAMCS, a feature-level fusion technique, named CSA is proposed, which increases the correlation between samples belongs to different classes and decreases the correlation between samples belongs to the same class. The recognition rate of MIRAMCS is better when compared to SvaS due to CSA. Experimental results prove the significance and validity of SvaS & MIRAMCS.

Chapter 6

Conclusion and Future Scope

6.1 Conclusions

This thesis investigates the reliable and privacy-preserving iris remote authentication techniques to solve the modify templates, intercept channel, and override comparator attacks of biometric recognition system.

In chapter 3, we assume that the server is "honest-but-curious" and proposed a privacy-preserving iris authentication system (PIAHC) using Fan-Vercauteren scheme. PIAHC avoids the rotational inconsistencies occured due to the head tilt of a person during the authentication phase results in the improvement of recognition accuracy. An algorithm to compute the Hamming distance between the encrypted reference and probe templates is proposed.

In chapter 4, a Blockchain-based Multi-instance Iris Authentication (BMIAE) method which combines Blockchain technology and ElGamal homomorphic encryption is proposed. Most of the existing template protection works based on homomorphic encryption rely on an implication that the server is "Honest-but-curious". Therefore, the compromise of such server fails to address override comparator attack of BAS results in the entire system vulnerability. This fact motivated us to design a method which not only provides the confidentiality of iris templates but also trust on the matching result. ElGamal encryption technique is used to achieve the confidentiality of iris templates and to perform matching in the encrypted domain. The Blockchain can emulate the functionality of an honest entity

as it is trusted for the correctness of execution and cannot be compromised. Additionally, the integrity of iris templates are also guaranteed by the Blockchain due to its immutability property.

Paillier homomorphic encryption is used for privacy-preserving and proposed two methods, namely secure and verifiable multi-instance iris authentication using public auditor (SviaPA), secure and verifiable multi-instance iris authentication using (SviaB). Reduction in the size of the iris template improves the overall computational performance. So, Auto-encoders are used to reduce the dimension size of iris template in SviaPA and SviaB. The correctness of comparator result is ensured by a public auditor in SviaPA and Smart contract running on a Blockchain in SviaB. The computational cost and time to authenticate a person are less in SviaB when compared to BMIAE. The limitations of Blockchain for biometrics like privacy and expensive storage cost are described in [119]. These limitations are also addressed in BMIAE and SviaB.

In chapter 5, a secure and verifiable machine learning based iris authentication method, namely SvaS is proposed. SvaS aims to achieve both privacy-preserving training and privacy-preserving classification of two classification algorithms, namely nearest neighbor and multi-class perceptron. SvaS includes a verification procedure to check the correctness of classification result returned by the cloud server. SvaS allows public verification i.e. anyone can verify the correctness of the computed result without the user's private information. A Privacy-preserving multi-instance iris authentication system is proposed to solve the modify templates and override comparator attacks of biometric recognition system. A feature-level fusion technique, Contradistinguish Similarity Analysis (CSA) which maximizes the pair-wise correlations and minimizes the between-class correlations is proposed. Fan-Vercauteran scheme is used to achieve the confidentiality of the fused iris templates. Polynomial factorization algorithm is used to check the correctness of the result returned by the cloud server.

All the techniques in chapter 3, 4, 5 satisfy all the requirements specified in the ISO/IEC 24745 standard. The proposed methods achieves better performance in terms of EER, d', KS-test. The proposed methods are experimented on publicly available iris databases and a comparative study of the proposed methods has been presented and discussed to

demonstrate their merits and capabilities.

6.2 Future Scope

- Most of the existing works assumed that the server is "Honest-but-curious". Therefore, the compromise of server results into the entire system vulnerability. So, a biometric remote authentication system needs to be developed in such a way that the system addresses not only privacy-preserving but also trust to the comparator result by maintaining the trade-off between time and cost.
- Multi-modal template protection schemes have to be developed to make use of benefits of multi-biometrics.
- Most of the iris template protection schemes have been evaluated on small and mid-size databases. However, these schemes have to be evaluated on large scale databases to prove their significance.
- The fully homomorphic encryption with more security and consumes less execution time need to be explored or developed, use it in BAS to make the template more secure.
- In BMIAE and SviaB, the encrypted reference templates are stored in server and only the hash values of encrypted reference templates are stored in the Blockchain. So, in future the template information has to be stored in the Blockchain itself in an optimal way.

Author's Publications

Journals:

1. Morampudi Mahesh Kumar, Munaga VNK Prasad, and U. S. N. Raju, "Privacy-preserving iris authentication using fully homomorphic encryption." *Multimedia Tools and Applications (Springer)*, pp. 1-23, 2020. DOI: https://doi.org/10.1007/s11042-020-08680-5 (In Press)
2. Morampudi Mahesh Kumar, Sowmya Veldandi, Munaga VNK Prasad, and U. S. N. Raju. "Multi-instance iris remote authentication using private multi-class perceptron on malicious cloud server." *Applied Intelligence (Springer)*, 2020. DOI: https://doi.org/10.1007/s10489-020-01681-9 (In Press)
3. Morampudi Mahesh Kumar, Munaga VNK Prasad, and U. S. N. Raju, "BMIAE: Blockchain-based Multi-instance Iris Authentication using Additive ElGamal Homomorphic Encryption." *IET Biometrics*, 9(4):165-177, 2020. DOI: https://doi.org/10.1049/iet-bmt.2019.0169
4. Morampudi Mahesh Kumar, Munaga VNK Prasad, Mridula Verma and U. S. N. Raju, "SvaS: Secure and Verifiable Machine Learning based Iris Authentication System using Fully Homomorphic Encryption." *Computers & Electrical Engineering (Elsevier*). (Revision Submitted)
5. Morampudi Mahesh Kumar, Munaga VNK Prasad, and U. S. N. Raju, "Secure and Verifiable Multi-Instance Iris Remote Authentication on untrusted Cloud Server." *IEEE Transactions on Information Forensics and Security*. (Communicated)

Conferences:

1. Morampudi Mahesh Kumar, Munaga VNK Prasad, and U. S. N. Raju. "Iris Template Protection using Discrete Logarithm." *In Proceedings of the 2018 2nd International Conference on Biometric Engineering and Applications*, pp. 43-49, 2018.

2. Morampudi Mahesh Kumar, Munaga VNK Prasad, and U. S. N. Raju. “Cancellable fingerprint template generation using rectangle-based adjoining minutiae Pairs.” *In Proceedings of the 2018 2nd International Conference on Biometric Engineering and Applications*, pp. 30-37, 2018.

Bibliography

[1] Nalini K Ratha, Jonathan H Connell, and Ruud M Bolle. An analysis of minutiae matching strength. In *International Conference on Audio-and Video-Based Biometric Person Authentication*, pages 223–228. Springer, 2001.

[2] Craig Gentry and Dan Boneh. *A fully homomorphic encryption scheme*. Stanford university Stanford, 2009.

[3] Lawrence O'Gorman. Comparing passwords, tokens, and biometrics for user authentication. *Proceedings of the IEEE*, 91(12):2021–2040, 2003.

[4] Anil K Jain, Patrick Flynn, and Arun A Ross. *Handbook of biometrics*. Springer Science & Business Media, 2007.

[5] Anil K Jain, Arun A Ross, and Karthik Nandakumar. *Introduction to biometrics*. Springer Science & Business Media, 2011.

[6] Salil Prabhakar, Sharath Pankanti, and Anil K Jain. Biometric recognition: Security and privacy concerns. *IEEE security & privacy*, 1(2):33–42, 2003.

[7] Anil K Jain, Arun Ross, and Salil Prabhakar. An introduction to biometric recognition. *IEEE Transactions on circuits and systems for video technology*, 14(1):4–20, 2004.

[8] Anil K Jain, Ruud Bolle, and Sharath Pankanti. *Biometrics: personal identification in networked society*, volume 479. Springer Science & Business Media, 2006.

[9] Ruud M Bolle, Jonathan H Connell, Sharath Pankanti, Nalini K Ratha, and Andrew W Senior. *Guide to biometrics*. Springer Science & Business Media, 2013.

[10] Nalini K. Ratha, Jonathan H. Connell, and Ruud M. Bolle. Enhancing security and privacy in biometrics-based authentication systems. *IBM systems Journal*, 40(3):614–634, 2001.

[11] Anil K Jain, Karthik Nandakumar, and Abhishek Nagar. Biometric template security. *EURASIP Journal on advances in signal processing*, 2008(113):1–17, 2008.

[12] Tsutomu Matsumoto, Hiroyuki Matsumoto, Koji Yamada, and Satoshi Hoshino. Impact of artificial" gummy" fingers on fingerprint systems. In *Optical Security and Counterfeit Deterrence Techniques IV*, volume 4677, pages 275–289. International Society for Optics and Photonics, 2002.

[13] Sujan TV Parthasaradhi, Reza Derakhshani, Larry A Hornak, and Stephanie AC Schuckers. Time-series detection of perspiration as a liveness test in fingerprint devices. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35(3):335–343, 2005.

[14] Athos Antonelli, Raffaele Cappelli, Dario Maio, and Davide Maltoni. Fake finger detection by skin distortion analysis. *IEEE Transactions on Information Forensics and Security*, 1(3):360–373, 2006.

[15] Hyung-Keun Jee, Sung-Uk Jung, and Jang-Hee Yoo. Liveness detection for embedded face recognition system. *International Journal of Biological and Medical Sciences*, 1(4):235–238, 2006.

[16] Patrizio Campisi. *Security and privacy in biometrics*, volume 24. Springer, 2013.

[17] Javier Galbally, Arun Ross, Marta Gomez-Barrero, Julian Fierrez, and Javier Ortega-Garcia. From the iriscode to the iris: A new vulnerability of iris recognition systems. *Black Hat Briefings USA*, 1, 2012.

[18] Javier Galbally, Arun Ross, Marta Gomez-Barrero, Julian Fierrez, and Javier Ortega-Garcia. Iris image reconstruction from binary templates: An efficient probabilistic approach based on genetic algorithms. *Computer Vision and Image Understanding*, 117(10):1512–1525, 2013.

[19] Guangcan Mai, Kai Cao, Pong C Yuen, and Anil K Jain. On the reconstruction of face images from deep face templates. *IEEE transactions on pattern analysis and machine intelligence*, 41(5):1188–1202, 2018.

[20] Emanuele Maiorana, Gabriel Emile Hine, and Patrizio Campisi. Hill-climbing attacks on multibiometrics recognition systems. *IEEE Transactions on Information Forensics and Security*, 10(5):900–915, 2014.

[21] Shreyas Venugopalan and Marios Savvides. How to generate spoofed irises from an iris code template. *IEEE Transactions on Information Forensics and Security*, 6(2):385–395, 2011.

[22] Karthik Nandakumar and Anil K Jain. Biometric template protection: Bridging the performance gap between theory and practice. *IEEE Signal Processing Magazine*, 32(5):88–100, 2015.

[23] Iso/iec 24745:2011 - information technology – security techniques – biometric information protection. https://www.iso.org/standard/52946.html. [accessed 05-October-2018].

[24] Vishal M Patel, Nalini K Ratha, and Rama Chellappa. Cancelable biometrics: A review. *IEEE Signal Processing Magazine*, 32(5):54–65, 2015.

[25] Stan Z Li. *Encyclopedia of Biometrics: I-Z.*, volume 2. Springer Science & Business Media, 2009.

[26] Ari Juels and Martin Wattenberg. A fuzzy commitment scheme. In *Proceedings of the 6th ACM conference on Computer and communications security*, pages 28–36, 1999.

[27] Ari Juels and Madhu Sudan. A fuzzy vault scheme. *Designs, Codes and Cryptography*, 38(2):237–257, 2006.

[28] Haiyong Chen and Hailiang Chen. A hybrid scheme for securing fingerprint templates. *International Journal of Information Security*, 9(5):353–361, 2010.

[29] Christian Rathgeb and Andreas Uhl. A survey on biometric cryptosystems and cancelable biometrics. *EURASIP Journal on Information Security*, 2011(1):1–25, 2011.

[30] Umut Uludag, Sharath Pankanti, Salil Prabhakar, and Anil K Jain. Biometric cryptosystems: issues and challenges. *Proceedings of the IEEE*, 92(6):948–960, 2004.

[31] Tanya Ignatenko and Frans MJ Willems. Information leakage in fuzzy commitment schemes. *IEEE Transactions on Information Forensics and Security*, 5(2):337–348, 2010.

[32] Marta Gomez-Barrero, Emanuele Maiorana, Javier Galbally, Patrizio Campisi, and Julian Fierrez. Multi-biometric template protection based on homomorphic encryption. *Pattern Recognition*, 67:149–163, 2017.

[33] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (CSUR)*, 51(4):1–35, 2018.

[34] Payal V Parmar, Shraddha B Padhar, Shafika N Patel, Niyatee I Bhatt, and Rutvij H Jhaveri. Survey of various homomorphic encryption algorithms and schemes. *International Journal of Computer Applications*, 91(8):1–7, 2014.

[35] Caroline Fontaine and Fabien Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security*, 2007(1):15–25, 2007.

[36] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[37] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, STOC '82, page 365–377, New York, NY, USA, 1982. Association for Computing Machinery.

[38] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[39] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[40] Josh Benaloh. Dense probabilistic encryption. In *Proceedings of the workshop on selected areas of cryptography*, pages 120–128, 1994.

[41] Tatsuaki Okamoto and Shigenori Uchiyama. A new public-key cryptosystem as secure as factoring. In *International conference on the theory and applications of cryptographic techniques*, pages 308–318. Springer, 1998.

[42] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International conference on the theory and applications of cryptographic techniques*, pages 223–238. Springer, 1999.

[43] David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. In *Proceedings of the 5th ACM conference on Computer and communications security*, pages 59–66, 1998.

[44] Ivan Damgård and Mads Jurik. A generalisation, a simpli. cation and some applications of paillier's probabilistic public-key system. In *International workshop on public key cryptography*, pages 119–136. Springer, 2001.

[45] Steven D Galbraith. Elliptic curve paillier schemes. *Journal of Cryptology*, 15(2):129–138, 2002.

[46] Andrew C Yao. Protocols for secure computations. In *23rd annual symposium on foundations of computer science (sfcs 1982)*, pages 160–164. IEEE, 1982.

[47] Tomas Sander, Adam Young, and Moti Yung. Non-interactive cryptocomputing for nc/sup 1. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, pages 554–566. IEEE, 1999.

[48] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In *Theory of Cryptography Conference*, pages 325–341. Springer, 2005.

[49] Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In *Theory of Cryptography Conference*, pages 575–594. Springer, 2007.

[50] Arun Ross and Anil K Jain. Multimodal biometrics: an overview. In *2004 12th European Signal Processing Conference*, pages 1221–1224. IEEE, 2004.

[51] Arun A Ross, Karthik Nandakumar, and Anil K Jain. *Handbook of multibiometrics*, volume 6. Springer Science & Business Media, 2006.

[52] Arun Ross and Anil Jain. Information fusion in biometrics. *Pattern recognition letters*, 24(13):2115–2125, 2003.

[53] Cees GM Snoek, Marcel Worring, and Arnold WM Smeulders. Early versus late fusion in semantic video analysis. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 399–402, 2005.

[54] Casia iris image database version 1.0. http://www.cbsr.ia.ac.cn/IrisDatabase.htm. [accessed 05-September-2017].

[55] Casia-v3 interval iris image database. http://www.cbsr.ia.ac.cn/IrisDatabase.htm. [accessed 05-September-2017].

[56] Ajay Kumar and Arun Passi. Comparison and combination of iris matchers for reliable personal authentication. *Pattern recognition*, 43(3):1016–1026, 2010.

[57] Yilong Yin, Lili Liu, and Xiwei Sun. Sdumla-hmt: a multimodal biometric database. In *Chinese Conference on Biometric Recognition*, pages 260–268. Springer, 2011.

[58] Christian Rathgeb, Andreas Uhl, Peter Wild, and Heinz Hofbauer. Design decisions for an iris recognition sdk. In *Handbook of iris recognition*, pages 359–396. Springer, 2016.

[59] Maneesh Upmanyu, Anoop M Namboodiri, Kannan Srinathan, and CV Jawahar. Blind authentication: a secure crypto-biometric verification protocol. *IEEE transactions on information forensics and security*, 5(2):255–268, 2010.

[60] Margarita Osadchy, Benny Pinkas, Ayman Jarrous, and Boaz Moskovich. Scifi-a system for secure face identification. In *2010 IEEE Symposium on Security and Privacy*, pages 239–254. IEEE, 2010.

[61] Yogachandran Rahulamathavan, Raphael C-W Phan, Jonathon A Chambers, and David J Parish. Facial expression recognition in the encrypted domain based on local fisher discriminant analysis. *IEEE Transactions on Affective Computing*, 4(1):83–92, 2012.

[62] Juan Ramón Troncoso-Pastoriza, Daniel González-Jiménez, and Fernando Pérez-González. Fully private noninteractive face verification. *IEEE Transactions on Information Forensics and Security*, 8(7):1101–1114, 2013.

[63] Jaroslav Šeděnka, Sathya Govindarajan, Paolo Gasti, and Kiran S Balagani. Secure outsourced biometric authentication with performance evaluation on smartphones. *IEEE Transactions on Information Forensics and Security*, 10(2):384–396, 2014.

[64] Georg M Penn, Gerhard Pötzelsberger, Martin Rohde, and Andreas Uhl. Customisation of paillier homomorphic encryption for efficient binary biometric feature vector matching. In *2014 International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–6. IEEE, 2014.

[65] Mohammad Haghighat, Saman Zonouz, and Mohamed Abdel-Mottaleb. Cloudid: Trustworthy cloud-based and cross-enterprise biometric identification. *Expert Systems with Applications*, 42(21):7905–7916, 2015.

[66] Masaya Yasuda, Takeshi Shimoyama, Jun Kogure, Kazuhiro Yokoyama, and Takeshi Koshiba. New packing method in somewhat homomorphic encryption and its applications. *Security and Communication Networks*, 8(13):2194–2213, 2015.

[67] Can Xiang, Chunming Tang, Yunlu Cai, and Qiuxia Xu. Privacy-preserving face recognition with outsourced computation. *Soft Computing*, 20(9):3735–3744, 2016.

[68] Changhee Hahn and Junbeom Hur. Efficient and privacy-preserving biometric identification in cloud. *ICT Express*, 2(3):135–139, 2016.

[69] Santosh Kumar, Sanjay Kumar Singh, Amit Kumar Singh, Shrikant Tiwari, and Ravi Shankar Singh. Privacy preserving security using biometrics in cloud computing. *Multimedia Tools and Applications*, 77(9):11017–11039, 2018.

[70] Motahareh Taheri, Saeed Mozaffari, and Parviz Keshavarzi. Face authentication in encrypted domain based on correlation filters. *Multimedia Tools and Applications*, 77(13):17043–17067, 2018.

[71] Vishnu Naresh Boddeti. Secure face matching using fully homomorphic encryption. In *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–10. IEEE, 2018.

[72] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012:1–19, 2012.

[73] Hui Zhu, Qing Wei, Xiaopeng Yang, Rongxing Lu, and Hui Li. Efficient and privacy-preserving online fingerprint authentication scheme over outsourced data. *IEEE Transactions on Cloud Computing*, to be published.

[74] Kai Zhou and Jian Ren. Passbio: Privacy-preserving user-centric biometric authentication. *IEEE Transactions on Information Forensics and Security*, 13(12):3050–3063, 2018.

[75] Joohee Lee, Dongwoo Kim, Duhyeong Kim, Yongsoo Song, Junbum Shin, and Jung Hee Cheon. Instant privacy-preserving biometric authentication for hamming distance. *IACR Cryptol. ePrint Arch.*, 2018:1214–1242, 2018.

[76] Mauro Barni, Giulia Droandi, Riccardo Lazzeretti, and Tommaso Pignata. Semba: secure multi-biometric authentication. *IET Biometrics*, 8(6):411–421, 2019.

[77] Shangwei Guo, Tao Xiang, and Xiaoguo Li. Towards efficient privacy-preserving face recognition in the cloud. *Signal Processing*, 164:320–328, 2019.

[78] Berkay Topcu and Hakan Erdogan. Fixed-length asymmetric binary hashing for fingerprint verification through gmm-svm based representations. *Pattern Recognition*, 88:409–420, 2019.

[79] Shengshan Hu, Minghui Li, Qian Wang, Sherman SM Chow, and Minxin Du. Outsourced biometric identification with privacy. *IEEE Transactions on Information Forensics and Security*, 13(10):2448–2463, 2018.

[80] Mrunal M Khedkar and SA Ladhake. Robust human iris pattern recognition system using neural network approach. In *2013 International Conference on Information Communication and Embedded Systems (ICICES)*, pages 78–83. IEEE, 2013.

[81] Himanshu Rai and Anamika Yadav. Iris recognition using combined support vector machine and hamming distance approach. *Expert systems with applications*, 41(2):588–593, 2014.

[82] Vivek Srivastava, Bipin Kumar Tripathi, and Vinay Kumar Pathak. Biometric recognition by hybridization of evolutionary fuzzy clustering with functional neural networks. *Journal of Ambient intelligence and humanized computing*, 5(4):525–537, 2014.

[83] K Saminathan, T Chakravarthy, and M Chithra Devi. Iris recognition based on kernels of support vector machine. *ICTACT J. Soft Comput.*, 5(5):889–895, 2015.

[84] Neda Ahmadi and Gholamreza Akbarizadeh. Hybrid robust iris recognition approach using iris image pre-processing, two-dimensional gabor features and multilayer perceptron neural network/pso. *Iet Biometrics*, 7(2):153–162, 2017.

[85] Md Fahim Faysal Khan, Ahnaf Akif, and MA Haque. Iris recognition using machine learning from smartphone captured images in visible light. In *2017 IEEE International Conference on Telecommunications and Photonics (ICTP)*, pages 33–37. IEEE, 2017.

[86] Mateusz Trokielewicz. Iris recognition with a database of iris images obtained in visible light using smartphone camera. In *2016 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA)*, pages 1–6. IEEE, 2016.

[87] Neda Ahmadi and Gholamreza Akbarizadeh. Iris tissue recognition based on gldm feature extraction and hybrid mlpnn-ica classifier. *Neural Computing and Applications*, 32:1–15, 2020.

[88] Alaa S Al-Waisy, Rami Qahwaji, Stanley Ipson, Shumoos Al-Fahdawi, and Tarek AM Nagem. A multi-biometric iris recognition system based on a deep learning approach. *Pattern Analysis and Applications*, 21(3):783–802, 2018.

[89] Neda Ahmadi, Mehrbakhsh Nilashi, Sarminah Samad, Tarik A Rashid, and Hossein Ahmadi. An intelligent method for iris recognition using supervised machine learning techniques. *Optics & Laser Technology*, 120:1–11, 2019.

[90] Muhammad Arsalan, Dong Seop Kim, Min Beom Lee, Muhammad Owais, and Kang Ryoung Park. Fred-net: Fully residual encoder–decoder network for accurate iris segmentation. *Expert Systems with Applications*, 122:217–241, 2019.

[91] Zijing Zhao and Ajay Kumar. A deep learning based unified framework to detect, segment and recognize irises using spatially corresponding features. *Pattern Recognition*, 93:546–557, 2019.

[92] Kuo Wang and Ajay Kumar. Cross-spectral iris recognition using cnn and supervised discrete hashing. *Pattern Recognition*, 86:85–98, 2019.

[93] Tianming Zhao, Yuanning Liu, Guang Huo, and Xiaodong Zhu. A deep learning iris recognition method based on capsule network architecture. *IEEE Access*, 7:49691–49701, 2019.

[94] Saša Adamović, Vladislav Miškovic, Nemanja Maček, Milan Milosavljević, Marko Šarac, Muzafer Saračević, and Milan Gnjatović. An efficient novel approach for iris recognition based on stylometric features and machine learning techniques. *Future Generation Computer Systems*, 107:144–157, 2020.

[95] Fadi N Sibai, Hafsa I Hosani, Raja M Naqbi, Salima Dhanhani, and Shaikha Shehhi. Iris recognition using artificial neural networks. *Expert Systems with Applications*, 38(5):5940–5946, 2011.

[96] Aparna Gale and Suresh Salankar. Analysis of iris identification system by using hybrid based pso classifier. In *ICDSMLA 2019*, pages 117–130. Springer, 2020.

[97] Maria De Marsico, Alfredo Petrosino, and Stefano Ricciardi. Iris recognition through machine learning techniques: A survey. *Pattern Recognition Letters*, 82:106–115, 2016.

[98] Claudio Orlandi, Alessandro Piva, and Mauro Barni. Oblivious neural network computing via homomorphic encryption. *EURASIP Journal on Information Security*, 2007:1–11, 2007.

[99] Mauro Barni, Pierluigi Failla, Riccardo Lazzeretti, Ahmad-Reza Sadeghi, and Thomas Schneider. Privacy-preserving ecg classification with branching programs and neural networks. *IEEE Transactions on Information Forensics and Security*, 6(2):452–468, 2011.

[100] Thore Graepel, Kristin Lauter, and Michael Naehrig. Ml confidential: Machine learning on encrypted data. In *International Conference on Information Security and Cryptology*, pages 1–21. Springer, 2012.

[101] Yogachandran Rahulamathavan, Suresh Veluru, Raphael C-W Phan, Jonathon A Chambers, and Muttukrishnan Rajarajan. Privacy-preserving clinical decision support system using gaussian kernel-based classification. *IEEE journal of biomedical and health informatics*, 18(1):56–66, 2013.

[102] Hui Zhu, Xiaoxia Liu, Rongxing Lu, and Hui Li. Efficient and privacy-preserving online medical prediagnosis framework using nonlinear svm. *IEEE journal of biomedical and health informatics*, 21(3):838–850, 2016.

[103] Ximeng Liu, Rongxing Lu, Jianfeng Ma, Le Chen, and Baodong Qin. Privacy-preserving patient-centric clinical decision support system on naive bayesian classification. *IEEE journal of biomedical and health informatics*, 20(2):655–668, 2015.

[104] Ping Li, Jin Li, Zhengan Huang, Tong Li, Chong-Zhi Gao, Siu-Ming Yiu, and Kai Chen. Multi-key privacy-preserving deep learning in cloud computing. *Future Generation Computer Systems*, 74:76–85, 2017.

[105] Ping Li, Jin Li, Zhengan Huang, Chong-Zhi Gao, Wen-Bin Chen, and Kai Chen. Privacy-preserving outsourced classification in cloud computing. *Cluster Computing*, 21(1):277–286, 2018.

[106] Raphael Bost, Raluca Ada Popa, Stephen Tu, and Shafi Goldwasser. Machine learning classification over encrypted data. In *NDSS*, volume 4324, pages 4325–4339, 2015.

[107] Xiaoqiang Sun, Peng Zhang, Joseph K Liu, Jianping Yu, and Weixin Xie. Private machine learning classification based on fully homomorphic encryption. *IEEE Transactions on Emerging Topics in Computing*, pages 1–13, 2018.

[108] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 1–23. Springer, 2010.

[109] Jiawei Yuan and Shucheng Yu. Privacy preserving back-propagation neural network learning made practical with cloud computing. *IEEE Transactions on Parallel and Distributed Systems*, 25(1):212–221, 2013.

[110] Qingchen Zhang, Laurence T Yang, and Zhikui Chen. Privacy preserving deep computation model on cloud for big data feature learning. *IEEE Transactions on Computers*, 65(5):1351–1362, 2015.

[111] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1–36, 2014.

[112] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[113] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin E. Lauter, Michael Naehrig, and John Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 201–210. JMLR.org, 2016.

[114] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1219–1234, 2012.

[115] Tong Li, Zhengan Huang, Ping Li, Zheli Liu, and Chunfu Jia. Outsourced privacy-preserving classification service over encrypted data. *Journal of Network and Computer Applications*, 106:100–110, 2018.

[116] Chen Wang, Andi Wang, Jian Xu, Qiang Wang, and Fucai Zhou. Outsourced privacy-preserving decision tree classification service over encrypted data. *Journal of Information Security and Applications*, 53:1–13, 2020.

[117] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, pages 365–377, 1982.

[118] Hervé Chabanne, Amaury de Wargny, Jonathan Milgram, Constance Morel, and Emmanuel Prouff. Privacy-preserving classification on deep neural network. *IACR Cryptol. ePrint Arch.*, 2017:35–53, 2017.

[119] Oscar Delgado-Mohatar, Julian Fierrez, Ruben Tolosana, and Ruben Vera-Rodriguez. Blockchain and biometrics: A first look into opportunities and challenges. In *International Congress on Blockchain and Applications*, pages 169–177. Springer, 2019.

[120] Oscar Delgado-Mohatar, Julian Fierrez, Ruben Tolosana, and Ruben Vera-Rodriguez. Biometric template storage with blockchain: A first look into cost and performance tradeoffs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[121] Oscar Delgado-Mohatar, Julian Fierrez, Ruben Tolosana, and Ruben Vera-Rodriguez. Blockchain meets biometrics: Concepts, application to template protection, and trends. *arXiv preprint arXiv:2003.09262*, 2020.

[122] AH Mohsin, AA Zaidan, BB Zaidan, OS Albahri, AS Albahri, MA Alsalem, and KI Mohammed. Based blockchain-pso-aes techniques in finger vein biometrics: A novel verification secure framework for patient authentication. *Computer Standards & Interfaces*, 66:1–14, 2019.

[123] Nigel P Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *International Workshop on Public Key Cryptography*, pages 420–443. Springer, 2010.

[124] Henning Perl, Michael Brenner, and Matthew Smith. Poster: An implementation of the fully homomorphic smart-vercauteren crypto-system. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS '11, page 837–840, New York, NY, USA, 2011. Association for Computing Machinery.

[125] Shai Halevi and Victor Shoup. Design and implementation of a homomorphic-encryption library. *IBM Research (Manuscript)*, 6:12–15, 2013.

[126] Léo Ducas and Daniele Micciancio. Fhew: bootstrapping homomorphic encryption in less than a second. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 617–640. Springer, 2015.

[127] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachene. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In *international conference on the theory and application of cryptology and information security*, pages 3–33. Springer, 2016.

[128] Hao Chen, Kim Laine, and Rachel Player. Simple encrypted arithmetic library-seal v2. 1. In *International Conference on Financial Cryptography and Data Security*, pages 3–18. Springer, 2017.

[129] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 409–437. Springer, 2017.

[130] Wei Dai and Berk Sunar. cuhe: A homomorphic encryption accelerator library. In *International Conference on Cryptography and Information Security in the Balkans*, pages 169–186. Springer, 2015.

[131] Yuriy Polyakov, Kurt Rohloff, and Gerard W Ryan. Palisade lattice cryptography library user manual. *Cybersecurity Research Center, New Jersey Institute ofTechnology (NJIT), Tech. Rep*, pages 1–40, 2017.

[132] Javier Ortega-Garcia, J Fierrez-Aguilar, D Simon, J Gonzalez, Marcos Faundez-Zanuy, V Espinosa, A Satue, I Hernaez, J-J Igarza, C Vivaracho, et al. Mcyt baseline corpus: a bimodal biometric database. *IEE Proceedings-Vision, Image and Signal Processing*, 150(6):395–401, 2003.

[133] Zvika Brakerski, Craig Gentry, and Shai Halevi. Packed ciphertexts in lwe-based homomorphic encryption. In *International Workshop on Public Key Cryptography*, pages 1–13. Springer, 2013.

[134] Nigel P Smart and Frederik Vercauteren. Fully homomorphic simd operations. *Designs, codes and cryptography*, 71(1):57–81, 2014.

[135] Michele Mosca. Cybersecurity in an era with quantum computers: will we be ready? *IEEE Security & Privacy*, 16(5):38–41, 2018.

[136] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):1–40, 2009.

[137] Wilson Abel Alberto Torres, Nandita Bhattacharjee, and Bala Srinivasan. Privacy-preserving biometrics authentication systems using fully homomorphic encryption. *International Journal of Pervasive Computing and Communications*, 11(2):151–168, 2015.

[138] Jung Hee Cheon, HeeWon Chung, Myungsun Kim, and Kang-Won Lee. Ghostshell: Secure biometric authentication using integrity-based homomorphic evaluations. *IACR Cryptology ePrint Archive*, 2016:484–509, 2016.

[139] Rohan Kulkarni and Anoop Namboodiri. Secure hamming distance based biometric authentication. In *Biometrics (ICB), 2013 International Conference on*, pages 1–6. IEEE, 2013.

[140] Rudresh Dwivedi, Somnath Dey, Ramveer Singh, and Aditya Prasad. A privacy-preserving cancelable iris template generation scheme using decimal encoding and look-up table mapping. *Computers & Security*, 65:373–386, 2017.

[141] P Punithavathi, S Geetha, and S Sasikala. Generation of cancelable iris template using bi-level transformation. In *Proceedings of the 6th International Conference on Bioinformatics and Biomedical Science*, pages 94–100. ACM, 2017.

[142] Morampudi Mahesh Kumar, Munaga VNK Prasad, and USN Raju. Iris template protection using discrete logarithm. In *Proceedings of the 2018 2nd International Conference on Biometric Engineering and Applications*, pages 43–49. ACM, 2018.

[143] Ramadan Gad, Muhammad Talha, Ahmed A Abd El-Latif, M Zorkany, EL-SAYED Ayman, EL-Fishawy Nawal, and Ghulam Muhammad. Iris recognition using multi-algorithmic approaches for cognitive internet of things (ciot) framework. *Future Generation Computer Systems*, 89:178–191, 2018.

[144] Yen-Lung Lai, Zhe Jin, Andrew Beng Jin Teoh, Bok-Min Goi, Wun-She Yap, Tong-Yuen Chai, and Christian Rathgeb. Cancellable iris template generation based on indexing-first-one hashing. *Pattern Recognition*, 64:105–117, 2017.

[145] Randa F Soliman, Mohamed Amin, and Fathi E Abd El-Samie. A double random phase encoding approach for cancelable iris recognition. *Optical and Quantum Electronics*, 50(8):326, 2018.

[146] Dongdong Zhao, Xiaoyi Hu, Jing Tian, Shengwu Xiong, and Jianwen Xiang. Iris template protection based on randomized response technique and aggregated block information. In *2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)*, pages 248–258. IEEE, 2018.

[147] Soubhagya Sankar Barpanda, Pankaj K Sa, Oge Marques, Banshidhar Majhi, and Sambit Bakshi. Iris recognition with tunable filter bank based feature. *Multimedia Tools and Applications*, 77(6):7637–7674, 2018.

[148] Debanjan Sadhya and Balasubramanian Raman. Generation of cancelable iris templates via randomized bit sampling. *IEEE Transactions on Information Forensics and Security*, 14(11):2972–2986, 2019.

[149] Randa F Soliman, Mohamed Amin, and Fathi E Abd El-Samie. Cancelable iris recognition system based on comb filter. *Multimedia Tools and Applications*, 79(3):2521–2541, 2020.

[150] Marta Gomez-Barrero, Christian Rathgeb, Guoqiang Li, Raghavendra Ramachandra, Javier Galbally, and Christoph Busch. Multi-biometric template protection based on bloom filters. *Information Fusion*, 42:37–50, 2018.

[151] Chetana Kamlaskar, Shubhangi Deshmukh, Suresh Gosavi, and Aditya Abhyankar. Novel canonical correlation analysis based feature level fusion algorithm for multimodal recognition in biometric sensor systems. *Sensor Letters*, 17(1):75–86, 2019.

[152] Gurjit Singh Walia, Shivam Rishi, Rajesh Asthana, Aarohi Kumar, and Anjana Gupta. Secure multimodal biometric system based on diffused graphs and optimal score fusion. *IET Biometrics*, 8(4):231–242, 2019.

[153] Shahzadi Farah, Younas Javed, Azra Shamim, and Tabassam Nawaz. An experimental study on performance evaluation of asymmetric encryption algorithms. In *Recent Advances in Information Science, Proceeding of the 3rd European Conf. of Computer Science,(EECS-12)*, pages 121–124, 2012.

[154] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[155] Yasi Wang, Hongxun Yao, and Sicheng Zhao. Auto-encoder based dimensionality reduction. *Neurocomputing*, 184:232–242, 2016.

[156] Christian Rathgeb and Christoph Busch. Cancelable multi-biometrics: Mixing iriscodes based on adaptive bloom filters. *Computers & Security*, 42:1–12, 2014.

[157] Evelyn Fix. *Discriminatory analysis: nonparametric discrimination, consistency properties*. USAF school of Aviation Medicine, 1951.

[158] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[159] Aman Kataria and MD Singh. A review of data classification using k-nearest neighbour algorithm. *International Journal of Emerging Technology and Advanced Engineering*, 3(6):354–360, 2013.

[160] Haneen Arafat Abu Alfeilat, Ahmad BA Hassanat, Omar Lasassmeh, Ahmad S Tarawneh, Mahmoud Bashir Alhasanat, Hamzeh S Eyal Salman, and VB Surya Prasath. Effects of distance measure choice on k-nearest neighbor classifier performance: A review. *Big data*, 7(4):221–248, 2019.

[161] Koby Crammer and Yoram Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3(1):951–991, 2003.

[162] Timothy LH Watkin, Albrecht Rau, Desiré Bollé, and Jort Van Mourik. Learning multi-class classification problems. *Journal de Physique I*, 2(2):167–180, 1992.

[163] Craig Gentry, Shai Halevi, and Nigel P Smart. Fully homomorphic encryption with polylog overhead. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 465–482. Springer, 2012.

[164] Mohammad Haghighat, Mohamed Abdel-Mottaleb, and Wadee Alhalabi. Discriminant correlation analysis: Real-time feature level fusion for multimodal biometric recognition. *IEEE Transactions on Information Forensics and Security*, 11(9):1984–1996, 2016.

[165] Christian Rathgeb, Benjamin Tams, Johannes Wagner, and Christoph Busch. Unlinkable improved multi-biometric iris fuzzy vault. *EURASIP Journal on Information Security*, 2016(1):1–26, 2016.

[166] Mousumi Sardar, Sushmita Mitra, and B Uma Shankar. Iris localization using rough entropy and csa: A soft computing approach. *Applied Soft Computing*, 67:61–69, 2018.

[167] Ali Noruzi, Mahmoud Mahlouji, and Ali Shahidinejad. Iris recognition in unconstrained environment on graphic processing units with cuda. *Artificial Intelligence Review*, pages 1–25, 2019.