An approach to authenticate magnetic stripe bank card transactions at point-of-sale terminals

KK Nair
20402333

Thesis submitted for the degree *Doctor Philosophiae* in Computer Engineering at the Potchefstroom Campus of the North-West University

Supervisor: Prof. ASJ Helberg

Assistant Supervisor: Johannes van der Merwe

November 2015

It all starts here ™

NORTH-WEST UNIVERSITY
YUNIBESITI YA BOKONE-BOPHIRIMA
NOORDWES-UNIVERSITEIT

Declaration

I, hereby declare that this thesis is a presentation of my original research work, conducted under the supervision of Prof. ASJ Helberg. Whenever contributions of others are involved, every effort has been made to indicate this clearly, with due reference to the literature. No part of this research has been submitted during the past, or is being submitted, for a degree or examination at any other University.

Kishor Krishnan Nair

November 2015

Editor's Certificate

NORTH-WEST UNIVERSITY
YUNIBESITI YA BOKONE-BOPHIRIMA
NOORDWES-UNIVERSITEIT
MAFIKENG CAMPUS

TEL: 018 389- 2168

ACADEMIC LITERACY UNIT

Dr.L.P Siziba

P.Bag X20146

Mamabatho, 2735

28th October 2015

To whom it may concern

Editorial Assistance for P.h.D Thesis

Declaration

Editorial intervention was restricted to: Language and Illustrations as well as Completeness and consistency as defined by the current South African standards for Editing Practice. Where the editor provided advice on structure, they gave examples only and did not undertake a structural re-write themselves. Material for editing or proofreading was submitted in hard copy, where an electronic copy was submitted to the editor, their mark up was done using tracking Changes and the file returned in a locked PDF format only. The decision to accept and implement changes suggested rests solely on the candidate.

The name of the editor and brief description of the service rendered has been provided below.

Acknowledged by:

Candidate's Name:	**Kishor Krishnan Nair**
I.D Number:	**7704166087189**
Student Number:	**20402333 NWU-Potchefstroom**
Thesis title:	**An approach to authenticate magnetic stripe bank card transactions at point of sale terminals.**

I declare that I have complied with the above conditions:

Signed: _____ Date: 28/10/2015

Editor's Name: DR LIQHWA P.SIZIBA

I declare that I have edited /proofread the thesis in compliance with the above conditions, as instructed when engaged by the candidate.

Signed: _____ Date: 28/10/2015

Acknowledgement

This thesis would not have been completed without Prof. ASJ Helberg who not only served as my supervisor, but also encouraged, supported, and advised me throughout the academic program. His guidance and advice were extremely valuable in contributing towards the successful completion of this research. I extend sincere thanks to my Assistant Supervisor and Research Group Leader Mr. Johannes van der Merwe at the Council for Scientific and Industrial Research (CSIR), South Africa, for offering all the support and valuable advice towards the completion of this study. I also extend my gratitude to the Potchefstroom Academic Administration for helping me throughout the curriculum.

I warmly thank Dr. Liqhwa P. Siziba of North-West University, Mafikeng campus for proof reading and correcting my thesis.

I am extending sincere thanks to my colleague Mr. Andre McDonald, at the CSIR for thoroughly reviewing and scrutinizing the thesis.

I wish to thank my mother Mrs. MS Santhakumari and my father Mr. PG Krishnan Nair. They showered me with unconditional love, provided everything that I wanted, gave me a good education, and prayed a lot for me.

I am extending sincere gratitude to my father-in-law Mr. Chandra Mohan Pillai and family, who have motivated, prayed, and helped me immensely throughout the study.

I owe loving thanks to my dear wife Manju, who has provided very valuable inputs, criticisms, and support in the completion of this study. I also owe loving thanks to my three little angels Keerthana, Kritha, and Krisha.

I always feel the presence of a great force that constantly guides,drives, and lifts me from difficult situations in life. To, the Almighty, Lord Ganesha, I dedicate this work to you.

Abstract

Magnetic stripe card technology has been deployed for more than five decades worldwide and is extensively used in banking. Data embedded in them are often relied upon as a benchmark for user authentication. As such reliance is placed upon them, it is surprising that they do not incorporate stringent security features and therefore attract the attention of criminals who compromise magnetic stripe cards for their illegal gain. Bank cards using magnetic stripe technology are being increasingly cloned or skimmed. Global statistics show that a fraudulent card transaction occurs every eight seconds and that cloning is the principal card fraud, which makes up approximately 37% of overall financial losses. Cloned magnetic stripe bank cards are extensively used at POS terminals and ATMs by criminals. POS terminals are one of the most commonly used payment transaction systems around the world. At the present moment, it is only the signature and PIN that prove the ownership of a magnetic stripe bank card. Even though chip cards are introduced as an extra security mechanism to avoid fraud, the fact that criminals can deliberately damage the chip and force the transaction to fallback to magnetic stripe defeats its intended security purpose. The result of all this fraud is that the original cardholders lose money unknowingly from their bank accounts. One way of enforcing a better security in POS terminals is by incorporating a biometric authentication system, preferably a Fingerprint Authentication System (FAS). This is due to the advantages and convenience that it offers above the other biometric counterparts. Although an FAS can prove the true ownership of a magnetic stripe bank card and can authenticate the transaction using it, this research recognizes existing vulnerabilities pertinent to FAS and biometric authentication systems in general. Hence, the usage of the conventional FAS may lead to severe security vulnerabilities. An FAS with robust security and acceptable recognition performance, at the present moment in time remains unclear and the development of such a system is vital. Thus, the proposal for an improved FAS is put forward to authenticate the transactions performed using magnetic stripe bank cards at POS terminals. The key underlying concept of the proposed system is a unique *One Time Fingerprint Template* which will be valid only for a single transaction session. The proposed FAS will be further verified, validated, evaluated, and criticised in order to illustrate the value added to this study.

Table of Contents

List of Figures

List of Tables

Acronyms and Abbreviations

The following is a list of acronyms and abbreviations that are used throughout this thesis.

ANSI	American National Standards Institute
APACS	Association for Payment Clearing Services
ATM	Automatic Teller Machine
BAS	Biometric Authentication Server
BIN	Bank Identification Number
BK	Biometric Key
BP	BAS Process
BPI	Bits Per Inch
BTT	Biometric Template Transformation
CFAS	Conventional Fingerprint Authentication System
CID	Card Identification Number
CVM	Cardholder Verification Method
DES	Data Encryption Standard
DFD	Data Flow Diagramming
DoS	Denial of Service
DS	Date Stamp
DUKPT	Derived Unique Key Per Transaction
EER	Equal Error Rate
EFT	Electronic Fund Transfer
EMV	Europay MasterCard Visa
EPB	Encrypted PIN Block
FAS	Fingerprint Authentication System
FASP	Fingerprint Authentication Security Protocol
FAR	False Acceptance Rate
FpVTE	Fingerprint Vendor Technology Evaluation
FRR	False Rejection Rate
FRVT	Facial Recognition Vendor Test
HSM	Hardware Security Module
IOTT	Intermediate One Time Template
ISO	International Organization for Standardization
MAC	Message Authentication Code

MSCD	Magnetic Stripe Card Data
OTP	One-time Password
OTT	One Time Template
PAN	Primary Account Number
PC	Personal Computer
PCI DSS	Payment Card Industry Data Security Standard
PED	Pin Entry Device
PG	Payment Gateway
PFAS	Proposed Fingerprint Authentication System
POS	Point-of-Sale
PSP	Payment Service Provider
PP	POS Process
SSADM	Structured System Analysis and Design Methodology
TCP/IP	Transmission Control and Internet Protocol
TLS	Transport Layer Security
TRSM	Tamper Resistant Security Module
TRN	Tokenised Random Number
TS	Time Stamp
TSN	Transaction Number
TT	Transformed Template
UP	User Process

1 Introduction

1.1 Introduction

Over the years, magnetic stripe card technology has been extensively used by the banking industry to facilitate the transactions of its account holders. It is widely used for performing Electronic Funds Transfer (EFT), sale, and cash withdrawal by various payment systems such as point-of-sale (POS) terminals, Automated Teller Machines (ATMs), and mobile phones. A magnetic stripe card secures data by altering the magnetism of minute iron-based magnetic particles embedded as a stripe on the card [1]. These cards are mainly used in electronic payments, the purchase of goods, and bill payment applications. As the usage of magnetic stripe bank cards has increased, crimes committed through them have also increased significantly. Therefore, it has become a worldwide problem.

Card cloning is the foremost crime performed using magnetic stripe bank cards and has grown as an epidemic. According to a study conducted by the ATM Industry Association in 2014, the financial loss due to card cloning crimes exceeds $2 billion a year [2]. The research paper “Skimming the Surface: How Skimmer Fraud Has Become a Global Epidemic” published by Darren R. Hayes of Pace University raised alarming figures globally as a result of card cloning [3]. The average cost of resolving a card cloning incident is estimated at approximately $50,000 [4]. Considering the heavy financial losses globally due to card cloning, it is essential to mitigate it.

POS terminals, where customer credit or debit cards are swiped for payment, is one of the most frequently used Electronic payment or E-payment systems in the developed world [5, 6]. Terminals are used in face-to-face transactions. A merchant swipes a customer’s magnetic stripe bank card through the terminal or keys-in payment information, and the terminal facilitates the rest of the transaction-processing [7]. Since payments through magnetic stripe bank cards in major businesses are facilitated

through POS terminals, it is vital to accurately authenticate the transactions performed using them. The motivation for the research is detailed in the next section.

1.2 Research motivation

The key motivation for this research is due to the following core issues or concerns that still remain as a question mark among the banking industry, payment card manufacturers, and cardholders. They are formulated as follows.

1.2.1 Increase in card cloning

Payment card crimes are growing at an alarming rate. This is mainly because of two factors: first is the high reward offered, and the second is the anonymity provided by modern technology in committing such fraud [8]. As a result, card cloning is increasing heavily in virtually every major city of the U.S., U.K., South Africa, India, China, Europe, Canada, Latin America, and in other parts of the world, and it has become an international problem [8]. Card cloning is the most serious security vulnerability in the financial sector, and it makes up approximately 37% of the overall monetary losses [9, 10]. The negative impact of card cloning is substantial for all stakeholders involved in payment systems, and it challenges the payment system's integrity. Further, it directly affects industry relationships, merchant behaviours, as well as consumer, and employee trust.

Card cloning can be described as a process whereby a genuine bank card's magnetic stripe data is copied on to a fake card. This cloned card can then be used for doing transactions at POS terminals and to make cash withdrawals at ATMs. The process whereby a card's magnetic stripe is copied is generally known as skimming [11]. The card is swiped through a skimming device analogous to a magnetic stripe reader on a POS terminal. POS terminals and ATMs are not able to differentiate between a cloned card and the original, as the data on both magnetic stripe cards are identical. Any type of bank card that has a magnetic stripe can be cloned, which includes debit cards, credit cards, and cheque cards [11].

Card cloning occurs typically at retail outlets that process bank cards for payments. A typical scenario is a dishonest employee who swipes a customer's bank card for a transaction, subsequently skims it in a small handheld electronic device that scans and

stores the card data from a magnetic stripe. Later on, the employee exchanges cloned bank cards to a criminal syndicate whom he or she is part of. These cloned cards will then be used for performing transactions. As a result, the original cardholder will ultimately be at loss [12, 13]. A fraudulent card transaction takes place every eight seconds and card cloning is the primary form of magnetic stripe card fraud [14]. Globally, card cloning is costing billions of dollars in losses to the payment industry [15]. A study from the Association for Payment Clearing Services (APACS) reveals that card cloning is a major issue for consumers and retail outlets [16].

1.2.2 Chip card abuse

Introduction of chip card technology has undoubtedly helped to alleviate security issues associated with magnetic stripe bank cards [17]. Chip cards are standardised based on the Europay, MasterCard, Visa (EMV) specifications [18]. Although EMV chip cards alleviated security issues associated with an EFT transaction to some extent, criminals have found new ways to hinder the chip card security. One scenario is that criminals clone an EMV chip card and damage or disable the chip. Subsequently, the transaction will fallback to magnetic stripe and will proceed as a normal magnetic stripe card transaction when processed at a POS terminal [14]. Thus, in effect, the extra security provided by the EMV chip card is nullified.

1.2.3 Loss of cardholder's money

Financial losses to cardholders, occurring as a result of card cloning crimes are very high. This attributes to billions of dollars; considering the huge volume of transactions happening every day using payment cards. Retailers worldwide experience $580.5 million in card fraud losses and spend $6.47 billion annually on fraud prevention [19]. According to the 2014 Nilson Report[1], the annual global financial losses due to credit card and debit card fraud equate to $11.2 billion [20]. Millions of cardholders worldwide are victims of card cloning crime, and there is a huge volume of incidents occurring every day where money is stolen from the bank accounts of cardholders.

[1] Nilson Report is a leading publication covering payment system statistics worldwide.

1.2.4 Banks reluctant to pay victims of card fraud

Bank payment cards are not fool proof, as several weaknesses have been identified and reported, since the introduction of the technology. Further, there have been extensive instances of fraudulent abuses over the past. In many cases, banks are unwilling to admit that their systems could be at fault and refuse to reimburse victims of what is arguably a fraud [21]. Card issuers are taking advantage of a grey area in the banking code, which leaves customers with heavy financial losses. As the financial crisis impacts the yearly turnover, the banking industry is progressively hesitant to compensate customers who have had the money illegally withdrawn from their accounts [22].

1.2.5 Cardholder verification not performed

Cardholder verification is a method, which is used to authenticate the cardholder and is known as CVM [21]. The authenticity of the cardholder is presently verified using the following methods:

- A POS terminal can request a cardholder's Personal Identification Number (PIN) for cardholder authentication before proceeding with a transaction. Although this security mechanism is in place, the majority of POS terminal applications in the field complete a transaction without performing PIN authentication.

- The cardholder's signature on the card can be used for authentication purposes. A merchant can compare the signature on the card with the signature on the sales slip to perform the cardholder verification. As the majority of merchants do not perform authentication of the cardholder through signature verification, this security mechanism is often bypassed [22]. Moreover, the signature on a newly cloned card can be manipulated by the criminal to match his or her signature. Hence, signature verification itself is vulnerable.

- A merchant can classify whether the cardholder is a male or a female by looking at the prefix of the cardholder's name field printed on the card. In majority of the cases, nobody reads the cardholder's name, which leads to a security vulnerability.

The motivational factors that were discussed in this section led to the formulation of a research problem statement, which is as follows: *The existing authentication mechanisms are not capable of establishing the true ownership status of a person who is using a magnetic stripe bank card at the POS terminal. This is the primary reason that allows a criminal to perform a transaction at a POS terminal using a cloned magnetic stripe bank card.* The research goal detailed in the next section discusses how this research alleviates the issues mentioned above.

1.3 Research goal

The need for automatic and precise personal identification has become essential for our exceedingly interconnected information world to run smoothly. Historical automatic personal identification techniques, which use 'something that you know,' such as PIN, or 'something that you have,' such as an ID card, are not adequate to meet the security requirements of sensitive electronic transactions. None of these techniques are capable of differentiating between an authorised person and an impostor, who deceptively gains the access privilege of an authorised person. This is the key reason why the use of cloned magnetic stripe bank cards is predominant in POS terminal fraud.

Even though cardholder authentication mechanisms are in place, they have failed to accurately authenticate transactions performed using magnetic stripe bank cards at POS terminals. This is the primary reason why card cloning is increasing at a rapid rate. Currently, it is only the signature and PIN that prove the ownership of a cardholder [23]. Due to the vulnerabilities present in these security mechanisms, the responsibility of identifying a stolen or copied card is implicitly transferred to the retailer. According to security experts, there is only a one in five chance that a terminal in a retail outlet will detect a cloned card [24]. If retailers are failing to check simple card details such as name and signature, there is little expectation that they will be able to identify a cloned card. Although chip cards are introduced as an extra security mechanism to avoid fraud, the fact that the transactions can still fallback to magnetic stripe makes it less secure. The result of all this fraud is that the original cardholder loses money from his or her bank account.

Implementation of a robust security mechanism, such as biometrics is indeed a great challenge for any application that is of a secret nature. At present, biometrics are not standardised as an official CVM by the banks, card manufacturers, and the payment transaction frameworks around the world [14]. This is mainly because of the fact that the legacy transaction infrastructure needs to be changed significantly in order to incorporate biometrics, and hence banks are reluctant to make this change in the short term [25]. The biometric CVM is also not included in the EMV specifications or magnetic stripe bank card specifications [21]. Due to these factors, biometrics technology has not found widespread use in the banking industry, with only a small number of banks implementing proprietary biometric systems [26]. This is also a major contributing factor to existing security weaknesses.

Despite claims from banks regarding the strength of security mechanisms surrounding the use of banking cards, several security vulnerabilities have been identified in banking infrastructure. These vulnerabilities have been successfully exploited by criminals, leading to instances of card cloning [26]. However, there remains an obligation from the bank to apply a reliable and strong user authentication mechanism before granting access to confidential information and restricted resources [25]. To achieve this, it is essential for the banking industry to consider and implement biometric authentication mechanisms in POS terminals that are convenient for the user as well as to the key entities in the transaction processing chain. At present, the only legally acceptable, fully automated, and mature biometric technique is the fingerprint identification technique, which has been used and accepted in forensics since the early 1970s [27].

Currently, the world market for biometric systems is estimated at $112 million and Fingerprint Authentication Systems (FASs) alone account for approximately $100 million [27]. FASs for civilian applications and physical access control are growing at a rapid rate [27]. Although the existing FASs offer a superior security when compared to the conventional authentication mechanisms (such as PIN or password), they are also susceptible to inherent biometric security vulnerabilities. The biometric vulnerabilities will be studied in detail in Chapter 2.

The existing authentication mechanisms in POS terminals fail to bind a transaction performed with a payment card to the user of the card. Hence, they are not able to establish a 100% ownership of a person who is performing the transaction. Therefore, a person using a cloned magnetic stripe bank card, and with knowledge of the PIN can circumvent the current POS authentication mechanism. Hence, it is clear that a strong biometric authentication mechanism must be combined in POS terminals to achieve the expected security. Thus, *the research goal is to incorporate a robust fingerprint biometric authentication mechanism in POS terminals to authenticate transactions performed using magnetic stripe bank cards. The fingerprint biometric authentication mechanism that is incorporated must be strong enough to address the research problem and at the same time must not lead to additional security vulnerabilities.*

The proposed solution to achieve the research goal is detailed in the next section.

1.4 Proposed solution

Criminals clone magnetic stripe bank cards and use them extensively to perform financial transactions at POS terminals. These transactions are considered sensitive, as they involve money, precious cardholder information, and critical financial data. Hence, it is of paramount importance to accurately authenticate individuals, who initiate transactions that involve magnetic stripe bank cards. This research attempts to investigate and incorporate a robust FAS, which can authenticate transactions performed using magnetic stripe bank cards, at POS terminals.

The proposed solution generates a unique fingerprint template of the card owner for each transaction session. Moreover, the proposed solution does not require the storage and transmission of the original fingerprint template of the card owner, which is the root cause of existing biometric security vulnerabilities. The fingerprint authentication phase biometrically authenticates if the card belongs to the correct person. If the fingerprint authentication is successful, then it is concluded that it is the true owner of the card who is performing the transaction therefore, it is genuine. On the contrary, if the fingerprint authentication fails, then it is ascertained that the card does not belong to its true owner therefore it is possible that the transaction could be performed using a cloned card. Hence, the proposed solution prevents the transaction

to proceed further from the POS terminal and saves the original card holder from losing money from his or her bank account.

This research delivers a proposed solution by following a specific research methodology, which is explained in the next section.

1.5 Research methodology

The methodology used in this research is a combination of mainly three approaches that are typically used in the problem solving phase of software engineering projects. They are the waterfall model, the prototyping model, and the qualitative model. The research starts with adapting the process of the waterfall model. The waterfall model is a systematic and sequential approach, in which the development of software is seen as progressing downwards through the phases of analysis, design, implementation, testing, and maintenance [28]. This research is planned to be completed in four phases.

Phase 1 starts with the feasibility analysis of the research, and the formulation of the research problem statement through a methodical literature survey. It addresses each entity in the research problem in detail. After a proper analysis and literature study in *phase 1*, the research progresses to *phase 2.* In *phase 2*, existing solutions to address the research problem are discussed and the motivation for a new system is identified from a list of key security objectives. Further, in *phase 2*, the new system is designed, and each component explained in detail.

In *phase 3*, the research adapts a combination of the waterfall model and the prototyping model. In the prototyping model, a functional prototype is used to model the proposed system [28]. The proposed model is thoroughly verified and validated, thus reverting to the waterfall model to ascertain if the research has indeed achieved its intended objectives laid out in *phase 2.*

The research will then step forward in the last phase, which is *phase 4* by adapting a test strategy. The test strategy used for this research is based on qualitative research methodology. Qualitative research is defined as “a process of inquiry with the goal of understanding a social or human problem from multiple perspectives; conducted in a natural setting with a goal of building a complex and holistic picture of the

phenomenon of interest" [29]. Qualitative research uses direct observation as one of the techniques for data collection and analysis [30]. In this approach, artifacts and photographs are typically collected as one of the methods for data validation [31]. By following this approach, in this study, traces, test results, screenshots, screen dumps, and plotted graphs are obtained by simulating the prototype. The test data, thus collected is further analysed and evaluated. A study is also conducted in *phase 4* to address the significance of the research, its limitations, and the scope for future work.

The following section explains terminology that recurs throughout this thesis.

1.6 Terminology

The terms that are frequently used throughout the thesis and their definitions are provided in Table 1.1 below.

Table 1.1: Terminology list

Terminology	**Definition**
POS terminal	A POS terminal is an electronic device that is used for capturing, verifying, and processing transactions using payment cards.
EFT	An EFT is the transfer or electronic exchange of money from one account to another, either within the same financial institution or across multiple institutions [32].
EMV Co	EMV Co manages, maintains, and enhances EMV® Integrated Circuit Card Specifications for chip-based payment cards and acceptance devices, including POS terminals and ATMs [18].
Biometrics	Biometrics is the science and technology of measuring and analysing biological data. In Information Technology, biometrics refers to technologies that measure and analyse human body characteristics, such as fingerprints, eye retinas and irises, voice patterns, facial patterns, and hand measurements for authentication purposes [33].
Card cloning	Card cloning can be described as a process whereby a genuine bank card's magnetic stripe is copied and then placed in a duplicate card. Card cloning is also known as skimming.
Magnetic stripe bank cards	Magnetic stripe bank cards are bank cards that are created based on magnetic stripe technology. Bank account information of cardholders is

	embossed on the magnetic stripe of these cards, thereby facilitating payments electronically.
Banking server	A banking server is typically a host machine that resides in a bank, with the purpose of processing transaction messages. It authenticates and processes transactions originating from various payment systems.
Authentication	Authentication is the process of establishing or confirming something or someone as authentic. Typical authentication examples involve, confirming the identity of a person using an identity card, a driver's license or a computer password.
FAS	FAS stands for Fingerprint Authentication System. It is a biometric authentication process based on finger templates.
FASP	FASP stands for Fingerprint Authentication System Protocol. It is the core communication protocol used in the FAS.
PIN	Personal Identification Number (PIN) is a unique number that is used in cases where the transaction needs to be authenticated for security purposes.
PAN	PAN stands for Primary Account Number. Although PAN can extend up to 19 digits, typically it is a 16-digit numeric code embossed on the face side of a bank card and it is also encoded in the magnetic stripe of a bank card. PAN is a composite number containing: the major industry identifier of the card issuer; an individual account identifier, which includes part of the account number; and a check digit that verifies authenticity of an account number [34].
CVM	CVM stands for Cardholder Verification Methods. CVM is a set of methods that are generally used in payment systems to authenticate the ownership of a cardholder.
Vulnerability	A vulnerability is defined as a part of a system that is easily exposed to damage [35].

The research plans and layout of this thesis are presented next.

1.7 Thesis layout

This thesis is organised into eight chapters.

Chapter 1 introduces the research, explains and motivates the research problem, identifies the research goal, proposes a solution to tackle the research problem, explains the research methodology, test strategy and clarifies the terminology used throughout the thesis.

Chapter 2 studies an existing POS transaction framework. It conducts a thorough security analysis and studies existing approaches to mitigate card cloning in the current framework. Biometric security based on fingerprints is reviewed and a feasibility study of fingerprint authentication is also conducted in its aim to address the research problem. This chapter further sheds light on biometric security issues and fingerprint security vulnerabilities.

Chapter 3 identifies entities in a fingerprint authentication system, and addresses its privacy and security concerns. The existing finger template protection schemes are examined and discussed in detail.

Chapter 4 proposes a novel FAS with its intended objectives to address the research problem. Moreover, the proposed system and its underlying security model are conceptualised and explained in detail.

Chapter 5 conducts an in-depth analysis of the proposed FAS to ascertain if it can indeed address the research problem. Individual entities in the system are designed using a structured system analysis and design methodology. The FASP behind the framework is designed and explained in detail.

Chapter 6 analyses various security protocol modeling tools and decides on ProVerif to measure the FAS. The ProVerif grammar is explained in detail. This chapter further derives a ProVerif model of the FAS. It then verifies and validates the proposed FAS based on the derived model.

Chapter 7 focuses on the evaluation of the proposed FAS. For this purpose, this chapter strategizes and implements a simulator for the proposed FAS in Matlab. The testing of the proposed FAS is performed extensively against the recognised Fingerprint Verification Competition (FVC) database. The test results are captured and the relevant graphs are plotted. Further, the results are analysed and the performance and usability aspects of the FAS are benchmarked.

Chapter 8 concludes the thesis by summarising the importance of this study. It describes the significance of this research and its specific contributions. The applications of this research, its limitations, and the scope for improvement are also provided in this chapter.

2

Scope of fingerprint authentication in a POS transaction processing framework to address the research problem

This chapter focuses on the scope of applying fingerprint authentication in a POS transaction processing framework to address the research problem. The chapter is structured as follows: the current POS transaction processing framework and its system security analysis are conducted in section 2.1 and 2.2 respectively. The existing approaches to mitigate card cloning and their pitfalls are addressed in section 2.3. In section 2.4, an overview of biometric security based on fingerprints is provided. A thorough analysis and feasibility study of biometric security using fingerprints is conducted in section 2.5. In section 2.6, the vulnerabilities associated with fingerprint biometrics are laid out and the chapter is concluded in section 2.7.

2.1 Existing POS transaction processing framework

This section presents a study of the existing POS transaction framework that is typically used in a payment environment, as shown in Figure 2.1. The POS transaction framework consitutes a cluster of technologies, which executes financial transactions through the electronic exchange of messages. For clarity purposes, only the core entities are represented in the figure (in specialised settings, further parties may well be involved). All key entities in the transaction chain and their roles are identified, as it is very much important to understand the information flow between them.

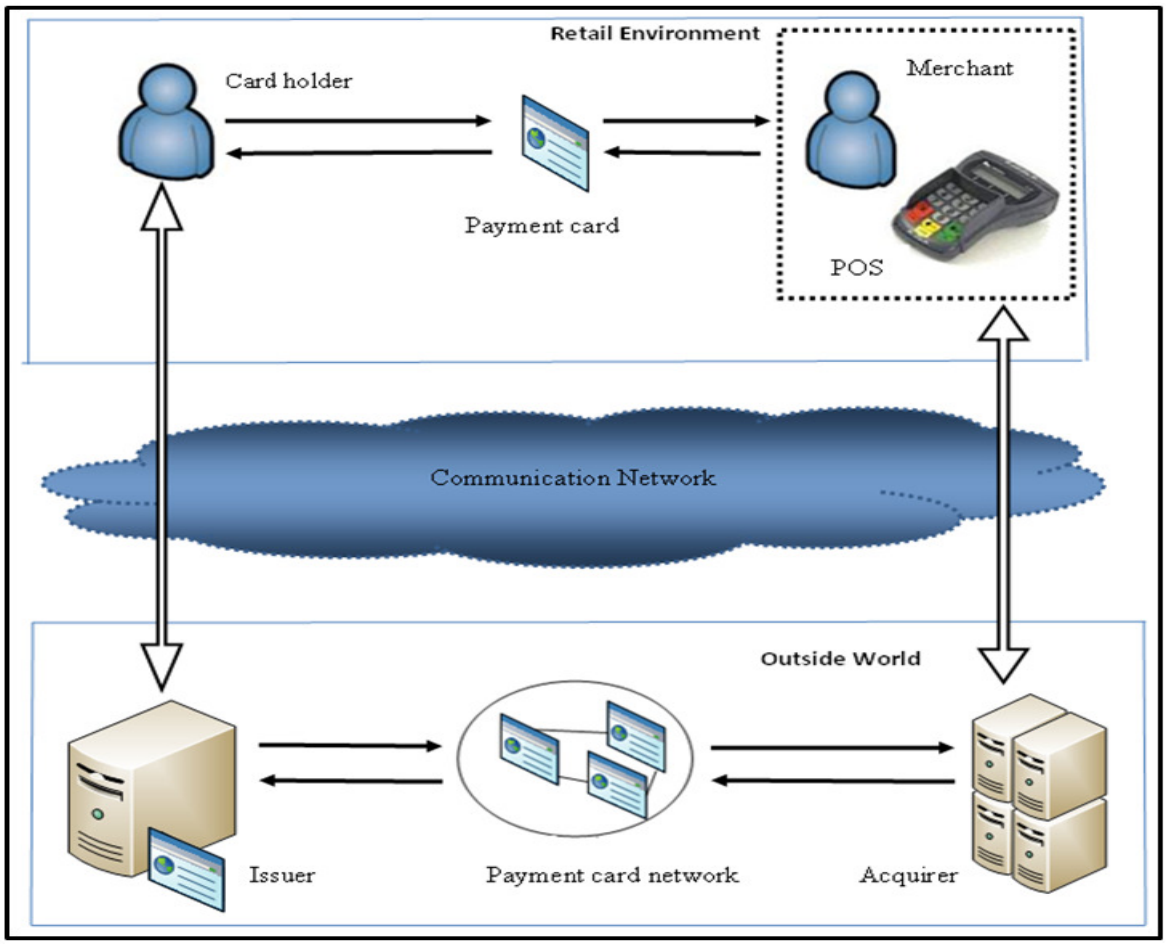


Figure 2.1: POS transaction processing framework [37]

Each component in the framework and their roles are explained as follows.

- **POS terminal**

  The POS terminal is the initial entry point in the transaction framework and is deployed in the merchant's till points. It is the most important entity within the transaction framework [6]. It captures the product data and starts with transaction processing by reading the cardholder data [7]. During the transaction processing, the cardholder may or may not be prompted for a PIN to establish the legitimacy of a transaction. After this step, the terminal communicates with a host entity such as an acquirer or an issuer to obtain authentication for the transaction. If the transaction is successful, a success slip is printed; otherwise, an error slip is printed. The card acceptor[2] accepts cards as a means of payment for goods or services. In payment systems, this may be a retailer, a service company or a

[2] In payment systems, card acceptor refers to an entity which reads the relevant cardholder data in order to for the purpose of processing transactions.

financial institution. Instead of accepting the card as a direct proof of payment, the acceptor may forward transaction information to an acquirer. The acceptor will accept a transaction authorisation from the acquirer as a guarantee for the payment. The security of the transaction information exchanged with the acquirer is important. Security features may include message authentication, PIN authentication, and implementation of security zones or a combination of all.

- **Payment card**

The payment card used for a transaction can be either a magnetic stripe or a smart card. Since the research focuses on magnetic stripe bank cards, only transactions using these cards will be considered. During a transaction, the magnetic stripe of the transaction card is swiped at the terminal [6]. The terminal reads the relevant cardholder data, such as the PAN and the card expiry date, and processes the transaction accordingly [7]. All magnetic stripe cards used in the payments industry have the same basic features. They all have a standard size and encode information such as the PAN, Bank Identification Number (BIN), expiry date, Card Identification Number (CID), and other card manufacturer details [36]. The card serves to identify the cardholder and the card issuer, and they may or may not agree on a secret PIN to be used during transactions. The transaction processing system has an obligation to maintain the PIN secrecy while moving the transaction from the cardholder to the card issuer. There are three tracks on the payment card. The ISO standard 7811, which is used by banking industry, specifies the tracks as follows [36]:

- Track 1 is 210 bits per inch (bpi), and accommodates 79 six-bit plus parity bit read-only characters.
- Track 2 is 75 bpi, and accommodates 40 four-bit plus parity bit characters.
- Track 3 is 210 bpi, and accommodates 107 four-bit plus parity bit characters.

The layout and contents of track 1, track 2, and track 3 of the payment card are illustrated in Figure 2.2, 2.3, and 2.4 of the appendix.

- **Acquirer**

  An acquirer is a bank or a financial institution that is accountable for its customer's transaction with the payment card network and acts on behalf of the merchant to process the transaction [37]. The acquirer sends transactions to the payment card network and is responsible for settling transactions on behalf of the merchant [38].

- **Payment card network**

  The payment card network is also known as the 'card interchange network' and is a group of financial entities that communicates to manage the processing, clearing, and settlement of bank card transactions [38]. The exact payment card network depends on the bank card used during the transaction. For example, if MasterCard is used, the transaction is forwarded to MasterCard network; if Visa is used, the transaction is forwarded to Visa network, and so on.

- **Issuer**

  The issuer is a bank or financial institution acting on behalf of the customer and is the entity responsible for issuing the bank card and maintaining the customer account [37]. The issuing bank authenticates the transactions that originate from the merchants' POS. The issuer approves or declines the transaction based on the customer's available funds [38]. It deals with the merchants acquiring bank and performs the steps to move the money from the customer's account to the acquirer.

A payment card transaction normally consists of two phases. They are explained in the following subsections.

2.1.1 Transaction flow

As illustrated in Figure 2.1, the transaction process commences when a payment card is presented to the POS terminal. The terminal records all the necessary transaction data and transfers the card data together with the transaction amount to the acquirer. The acquirer passes the transaction data to the issuer via the payment card network. The issuer verifies the account status in the database and responds to the acquirer, who then transfers the authorization code to the terminal [38]. The process of accumulating the funds from the issuing bank and settling the merchant can only start after the transaction details are

transferred to the acquirer. This process is known as *clearing and settlement* or simply *settlement*, which is elaborated in the next section.

2.1.2 Clearing and settlement

In this process, after the acquirer obtains the transaction details, it transfers the information to the appropriate payment card network, by which the transaction data is routed to the respective issuers. The issuer subsequently charges the cardholders for the transaction amount and remits funds less the issuer's fee through the network to the acquirer. The acquirer afterwards deducts the fees for the issuer, the network, and itself. It then reimburses the rest of the fund to the merchant's account within 24 to 72 hours [38]. The next section conducts a system security description of the existing transaction framework.

2.2 System security description

Security in the POS payment systems implies the protection of the entities that are participating in the transaction framework and safeguarding the information against unauthorised or illegal access and use. The safety of the information and assets entrusted to any POS transaction framework is solely dependent upon the degree of security that it is offering. It must protect data and funds against abuse, theft, and loss at all times. The users of the framework should be guaranteed that transactions will be carried out securely and only based on their instructions. The security in the POS payment systems revolves around the following core security principles [39]:

- Confidentiality
- Authenticity
- Integrity
- Tamper proofing
- End-to-end security

Each of these security principles is further explained below.

- **Confidentiality**

Payment transactions must be kept confidential and financial matters are to be dealt with confidentiality; ensuring that the right level of information is only given to the correct entities. Each entity must have access to the necessary information to complete their

respective responsibilities, but must not have access to information that would challenge the transaction confidentiality.

- **Authenticity**

Authenticity guarantees that each transaction is essentially conducted using a legitimate card, by a legitimate cardholder, on a legitimate POS terminal, under the control of a legitimate merchant. The whole payment system must enforce transaction authenticity.

- **Integrity**

The payment system must ensure the integrity of each transaction, which implies that any alteration of the content of any information associated with a transaction must be identified, and that the transaction should be rejected.

- **Tamper proofing**

In order to ensure that all transaction information is kept confidential, the POS industry developed the concept of tamper proofing. Once a POS terminal is manufactured, and its firmware and keys are injected, it cannot be tampered with [40]. If any tamper attempt occurs within a terminal, such an attempt will be identified. The terminal then responds by deleting its content, making it unusable. This is called tamper responsiveness.

- **End-to-end security**

The need for secure transactions, along with the concept of tamper proofing is the foundation of the end-to-end security concept. The security of a payment system implies that all its entities must be secure. However, threats can occur from any point in a transaction framework. A multitude of technologies and procedures are already implemented to incorporate the above security concepts in the current financial transaction framework and its underlying financial network. The existing system security mechanisms will be discussed in the following subsections.

2.2.1 PIN security

The PIN is a security mechanism used by the issuing bank to verify the identity of its account holders. When a PIN is entered by the cardholder at a POS terminal, the PIN and the PAN number along with other transaction details are sent to the issuing bank or to an authorised entity for verification. To protect the PIN during transit, it is encrypted into a PIN block by the POS terminal using encryption keys [39]. The resulting Encrypted PIN Block (EPB) is sent for verification to the issuing entity. The EPB has to pass through

each entity in the transaction framework before it reaches the destination and will be ultimately verified at the destination.

2.2.2 Security zones

Security between different entities in a transaction chain is classified into security zones and it is crucial in implementing end-to-end security. A security zone is an area within a network occupied by a group of security systems and components aimed to protect sensitive information [39, 40]. Each security zone in a payment system typically implements its security using a Hardware Security Module (HSM) which is a Tamper-Resistant Security Module (TRSM) that protects PIN and encryption keys [39, 40]. The purpose of each security zone along the transaction chain is to decrypt sensitive information such as EPB in order to make sure that it is not altered. If needed, each security zone will re-format and re-encrypt the sensitive information before routing it to the next entity, thereby achieving an end-to-end transaction security [39, 40].

2.2.3 Message security

In addition to securing the customer PIN and implementing the security zones, all the sensitive financial transaction messages that flow between different entities also need to be secured. The message encryption in the current transaction framework is widely implemented using the Derived Unique Key Per Transaction (DUKPT) algorithm specified by the ANSI X9.24 standards. DUKPT is basically a key management technique which uses a unique key for each transaction, and averts the release of some past key used by the transaction-originating TRSM [39]. The transaction messages are also secured using the Message Authentication Code (MAC) algorithms [39]. The MAC is a cryptographic hash calculated from part or the entire transaction message using a secret MAC key [40].

In addition to the above security mechanisms, it is a security requirement that the keys used for PIN, DUKPT, and MAC encryption are to be injected to various entities in a secure manner. The key injection process is done in a highly secure and trusted environment known as the trusted center. This section conducted a system security description of the current POS transaction framework; the next section examines the existing approaches to mitigate card cloning.

2.3 Existing approaches to mitigate card cloning

The importance of information security in payment systems cannot be overvalued as a security breach can result in significant financial loss and irreparable damage to a company's reputation. Criminals are targeting merchants using susceptible payment applications and exploiting the vulnerabilities to extract critical transaction data [41]. Card cloning stands as one of the highly profitable criminal activities that are committed in the financial sector [41]. This makes it more attractive, and hence it is becoming extremely difficult to prevent it. Card cloning allows the capture of immense volumes of account details in a small time frame, with little risk of detection [42]. Trustwave is a company that scrutinizes payment card compromises for companies such as Visa, MasterCard, and American Express. They have conducted 220 studies globally involving the information compromises in 2013/2014. The vast majority of the cases came down to the flaw in POS terminals [43].

The following subsections discuss the existing approaches that are in place to mitigate card cloning, and analyse their effectiveness.

2.3.1 Migrating from magnetic stripe bank cards to smart cards

Migration implies the phasing out of magnetic stripe bank cards that are in use today and reissuing all existing customers with smart cards. There are more than 3 billion magnetic stripe bank cards in circulation around the world today, which is the primary challenge faced by the migration process. It is unlikely that the process of migration will be completed in the short term [44]. The rate of adoption of smart card technology has been slow so far, and major markets like the U.S. are still to adopt this technology. One of the biggest bottlenecks is the cost. It is estimated that replacing the existing POS terminals with terminals that are capable of processing a smart card transaction will cost tens of millions of dollars in the U.S. alone [44]. Further, card-issuing banks will need to spend millions of dollars to upgrade their networks and internal systems to cater for smart card transactions [44].

2.3.2 Diebold's ATM security protection suite

This product consists of anti-cloning packages coupled with monitoring services to provide effective countermeasures against card cloning. It facilitates five levels of protection to guard against the sophisticated card cloning attacks and financial institutions are provided with an option based upon the level of protection that they need [45]. Level one provides basic protection and includes ATM card reader security features specially designed to dissuade the cloner attachment. Level two offers cloning detection technology that generates an alert that is directed either to the branch alarm system or to the ATM network monitoring system when a fraudulent device has been added to the ATM. Level three and level four incorporates cloning countermeasures by emitting an electromagnetic field to interfere with a cloner's ability to capture a magnetic stripe data in card readers and thus helps in preventing the capture of card information.

2.3.3 MagnePrint®

MagnePrint® is a dynamic card authentication technology that determines the originality of the card, based on the unique physical properties of the magnetic stripe. When the card is first issued, the card issuer transforms the digitized original MagnePrint to a 54 byte string. It is known as the *Reference MagnePrint* and is stored in an Authorisation Server (AS). The MagnePrint technology works with special readers that recover the encoded track data and the MagnePrint from the AS. When a card is read, the encoded card data, the MagnePrint and the transaction details are sent to the AS for verification. The MagnePrint captured during this time is known as the *Transaction MagnePrint*. The unique features about a Transaction MagnePrint is that it changes dynamically, and that the changes are unpredictable. The chances of obtaining two identical 54 byte Transaction MagnePrints from a single card are about 1 in 100 million [46]. Hence, during the verification phase, a Transaction MagnePrint identical to the one previously used will be rejected.

2.3.4 PCI DSS compliance

Security mechanisms that are generic to the financial transactions are implemented according to the standards directed by the PCI DSS council. The standard mandates compliance in many aspects, including secure networks, cardholder data protection, access control, vulnerability management, security assessments, and reporting [41].

2.3.5 Proprietary biometric authentication frameworks

In academic research, there are two major proprietary authentication frameworks proposed by the research community to biometrically validate financial transactions. The first is a biometric framework for the ATM, and the second is a remote biometric framework using smart cards. The biometric framework for the ATM was proposed by Hammed Lasisi and Adedeji Ajisafe and the smart card biometric framework was proposed by Chun-Ta Li and Min-Shiang Hwang [47, 48].

Since the above subsections from 2.3.1 to 2.3.5 have already discussed the existing solutions, the following paragraphs will therefore, conduct a qualitative comparative study of each solution.

Although the smart card seems a viable solution, the drawback is that a smart card costs about 100 times more than a magnetic stripe card [46]. In addition, a large investment has been made in the current magnetic stripe card system and the payment terminals. It is, therefore, unlikely that the existing payment infrastructure will be replaced in a short term. Another issue, as pointed out in Chapter 1, is that, criminals clone the magnetic stripe data of the smart card and damage or disable the chip intentionally in the cloned smart card. As a result, when a cloned smart card is processed at a POS terminal, each transaction will fallback to the magnetic stripe (i.e. each transaction will be processed as a normal magnetic stripe card transaction). In this manner, the smart card security mechanisms are effectively bypassed.

In order for the MagnePrint® solution to be practical and function in the operational environment, the entire card processing devices must be replaced and issuers must agree to record and share their card's magnetic data signatures. It is also a mandatory requirement that all merchants must agree to use POS terminals that have the ability to read the magnetic signature of the card. The fulfillment of these requirements leads to extra overhead, cost, and inconvenience, thereby rendering this solution infeasible.

The solution provided by the Diebold is only intended for ATMs and does not address the card cloning issue in POS terminals. The PCI DSS council enforces financial institutions and payment networks to implement the requirements which are proposed in its standards. These requirements are non-trivial to be implemented due to their complexity in both

technical and organisational terms. In order to comply, it is necessary to perform continuous assessments of the standard's security programs, which is often regarded as a burden to many merchants. In addition, it has been observed that many merchants, acquirers, and service providers are not conforming to the PCI DSS standards. As reported by Visa, only 22% percent of its largest merchants were compliant, in addition to smaller merchants with tight budgets and resources [46]. The main issue with this standard is that it does not address the card cloning issue and hence even if the financial institutions comply with the standard, the card cloning issue will still prevail.

The following deductions were also made after carefully studying the proprietary biometric authentication frameworks proposed by various researchers. It was observed that the authentication process followed in the biometric ATM framework proposed by Hammed Lasisi and Adedeji Ajisafe is prone to high False Rejection Rate (FRR), and that the biometric authentication protocol used in the framework is weak (FRR will be clarified under section 2.5) [47]. Furthermore, during enrollment, the original finger templates are captured and stored in the database. This is highly risky because if the finger template database gets compromised, all the finger templates are lost forever. Apart from this, the message exchanges within the framework are not encrypted and is a huge privacy and security concern. Therefore, the usage of this framework can lead to future biometric security vulnerabilities and significant user inconvenience.

The smart card biometric framework proposed by Chun-Ta Li and Min-Shiang Hwang has not addressed any specific use case of their model or mentioned the particular biometric modality such as fingerprint or face recognition that needs to be used in their scheme. Moreover, their scheme requires a smart card to perform the user authentication and a bank card to perform the transaction, should the system be deployed in the financial environment [48]. Thus, the proprietary biometric authentication frameworks are highly inconvenient and impractical to be used in the existing POS transaction processing framework. Furthermore, the integrity and security of the templates can be guaranteed only to a limited extent by these frameworks.

Card cloning fraud is continuing to evolve, with criminals devising increasingly sophisticated means to circumvent new countermeasures. There is as yet no clear and consistent set of industry-wide security standards for the protection of payment systems

against this fraud [49]. The root cause behind the card cloning issue is the remote nature of the transaction. In the current transaction scenario, the individual is at the remote end of a communication channel and can be authenticated only by weak security tokens that they possess, such as a password or a PIN. Payment systems should be capable of achieving robust user authentication to address card cloning, especially in an online environment. This can be only achieved by the use of biometric techniques, which will add top class security to the payment card transactions [49]. The next section will be conducting an extensive study of biometric security based on fingerprints in its aim to address the research problem.

2.4 Biometric security based on fingerprints

Biometric authentication based on fingerprints is believed to be the most convenient, efficient, distinctive, cost-effective, and a popular technique used in building robust security systems [27]. The following paragraphs will focus on the various aspects of fingerprint biometrics.

The skin present on the fingers is rough and is different from other areas of the body. It consists of raised sections known as ridges and is not continuous between the sides. Instead, they may curve, end (in which case the ridges are known as endings), or transform into two or more ridges (referred to as bifurcation). Minutiae are defined as those points of a fingerprint where the ridges become bifurcations and endings. They are the most discriminating and reliable features of a fingerprint. Furthermore, the amount of information that needs to be captured and stored for minutiae based techniques is smaller and the processing time is shorter than other techniques [50]. These unique features form the basis of any system using fingerprint comparison techniques for identification and verification purposes [51, 52, 53].

A finger template is a digital representation of an individual's fingerprint characteristics, containing information extracted from a fingerprint sample. Finger templates are compared with one another in a fingerprint recognition system [54]. Fingerprint identification is based upon finger template matching followed by the detection of minutiae characteristics [53]. Typically, a single rolled fingerprint contains more than 100 identification points that can be used for identification purposes [53]. In a study aimed at

quantifying the uniqueness of fingerprints, the U.S., Federal Bureau of Investigation (FBI), constructed a mathematical model of a fingerprint based on 50,000 distinct sample fingerprints. This model was compared with 50,000 other fingerprints. The study revealed that it was statistically nearly impossible (one in 10 million) for two fingerprints to agree on more than four minutiae characteristics [55]. The following list gives a comprehensive list of the advantages of using fingerprints.

- Subjects have multiple fingers.
- Fingerprint acquisition technology is easy to use, with some training.
- The enrollment systems require little memory.
- Large fingerprint databases already exist, which facilitates background checks.
- Fingerprint technology has proven effective in many large-scale systems over years of use.
- Fingerprints are unique to each finger of each individual, and the ridge arrangement remains permanent during one's lifetime.

The next section conducts the feasibility study of a Fingerprint Authentication System (FAS).

2.5 Feasibility study of an FAS

The use of biometric systems is becoming more wide spread, due to their reliability and robustness. FASs, among others, are the most popular and widely used type of biometric system [56]. It involves the presentation of a fingerprint for querying, comparing the presented fingerprint sample to a stored template and determining whether the individual has made a legitimate claim [57]. This section aims in criticizing the FAS by analysing its advantages and disadvantages, performance and reliability, error rate and security issues. It weighs various attributes and evaluates FAS in its effectiveness and capability to establish the unique identity of an individual in its aim to address the research problem. The following subsections analyse each aspect in detail.

2.5.1 Advantages

Fingerprints are widely used in biometrics research studies as well as in commercial applications [58]. The choice of FAS over other solutions overcomes religious and cultural barriers such as the exclusion of women's facial recognition in Muslim countries like Saudi Arabia [59]. Fingerprints are relatively stable throughout one's lifetime, and

updating of finger templates may only be needed in cases of major trauma to fingers. However, other systems such as face recognition systems are likely to require an update at least every 10 years [59].

2.5.2 Performance and reliability

The biometric database typically contains millions of enrollment records and hence the performance and reliability required of any biometric authentication system are very stringent. Therefore, the capture of biometric images needs to be quick, require little or no operator assistance, and at the same time, the system should generate images of sufficient quality. Current large-scale FASs are minutiae based and there are also systems that use the entire fingerprint pattern [58]. The performance and reliability considerations of fingerprint authentication systems are further elaborated in the following subsections.

2.5.2.1 Identification accuracy

The accuracy of fingerprint identification is based on a number of performance measures, which are as follows.

- ***False Acceptance Rate (FAR):*** The FAR measure is the likelihood of a person's biometric template being positively matched to the template of another person in the database [60]. In other words, the FAR is the rate at which a biometric system authenticates an unauthorized person or an imposter.
- ***False Rejection Rate (FRR):*** The FRR measure is the likelihood of a person's biometric template not being positively matched to his or her biometric template stored in the database [60]. In other words, the FRR is the rate at which a biometric system rejects an authorized person (i.e. the individual is not authenticated).
- ***Equal Error Rate (EER):*** EER is the rate at which the FAR and FRR are equal. The accuracy of a biometric system is usually measured by its EER. In general, a system with a low EER is more accurate than a system with a higher EER [60].
- ***Threshold:*** The threshold represents the degree of similarity required between the captured biometric template and the stored template before a match is declared between the two templates. Biometric implementations can realize a trade-off between the FAR and the FRR by changing the value of the threshold [60].

2.5.2.2 FAR/FRR analysis

At initial enrollment, there will be times when the images presented by the user and obtained by the system are not of sufficient quality. For example, in the case of fingerprint recognition, false acceptance or false rejection may occur due to dirty or dry fingers, which can be remedied by the user washing his or her hands, or by applying hand cream. In other cases, there may be fundamental reasons that make it difficult or impossible for a user to present his or her fingerprints to the reader in the normal way. For instance, users with arthritis may find it very hard to present flat impressions of their fingerprints on a standard capture device, and they may need to be enrolled one finger at a time on a different device. The performance studies indicate that for a single matching attempt against a single finger, reliable fingerprint systems are able to achieve an FAR of 1 in 100,000 with an FRR of approximately 1 in 100. These results are based on annual Fingerprint Verification Competitions [61].

2.5.2.3 Error rate

As mandated by the Enhanced Border Security Act and the U.S. Patriot Act, NIST conducted the Fingerprint Vendor Technology Evaluation (FpVTE) and Facial Recognition Vendor Test (FRVT) to evaluate the accuracy of FASs and facial recognition systems. These tests were designed to assess the capability of fingerprint and facial recognition systems to meet requirements for both small scale and large scale real-world applications. FpVTE consists of multiple tests performed by various combinations of finger template enrollment [62]. Similarly, FRVT consists of numerous tests performed using different combinations of facial images [63]. The following entries summarise the performance comparison of the fingerprint and facial recognition system documented in FpVTE and FRVT.

“The most accurate facial recognition systems have 90.3% correctness at 1% FAR and the most accurate FASs have a 99.9% correctness at 1.0% FAR” [62, 64].

The above test results clearly show that FASs are more accurate than facial recognition systems. Even though the FAS can enhance user convenience and reinforce security, it is

also susceptible to various types of threats that are inherent in the biometric security systems. They are elaborated under the fingerprint security vulnerabilities.

2.6 Fingerprint security vulnerabilities

In general, a biometric system is susceptible to a variety of attacks aimed at undermining the integrity of the biometric authentication process, and FAS is also vulnerable to these attacks. The attacks are intended to either compromise the security afforded by the system or to inhibit the normal functioning of the system. These are elaborated from an FAS perspective, which is as follows [64].

- ***Circumvention:*** Circumvention involves the creation of an artifact of an original biometric representation. Typically, in this attack an intruder first gain access to the FAS using an artifact and cross-examines sensitive data such as the financial data pertaining to a legitimately enrolled user. Apart from violating the privacy of the enrolled user, an intruder can also alter sensitive data.
- ***Repudiation:*** In this attack, a legitimate user can access the privileges offered by an application and afterwards claim that an intruder circumvented the system. For example, an accountant can manipulate the financial records of a company and then deny responsibility by claiming that an intruder could have possibly stolen the finger templates.
- ***Covert acquisition:*** An attacker may secretly obtain the raw finger templates of a user to access the system. For example, the latent fingerprints of a user may be picked up from an object and later used to build a digital or physical artifact of that user's finger.
- ***Collusion:*** In this attack, a user with super-user privileges can intentionally alter system parameters to allow intrusions.
- ***Coercion:*** In this attack typically a legitimate user is forced, for example, at gunpoint to provide system access to the attacker.
- ***Denial of Service (DoS):*** An attacker may overwhelm the system resources to the point where legitimate users desiring access to a service will be denied access. For example, a server that processes access requests can be flooded with a large number of bogus requests, thereby overloading its computational resources and preventing valid requests from being processed.

The vulnerabilities associated with the FAS are summarised as follows:

1. The most relevant vulnerability is that once a finger template is compromised, it cannot be reissued updated or destroyed [64].

2. Once a biometric modality is chosen, it can be used to access different systems. This means that if it is compromised, the attacker will have access to all the accounts, services, and applications. According to the security expert Bruce Schneier, biometrics are unique identifiers, but they are not secrets. "You leave your fingerprints on everything you touch and your iris patterns can be observed anywhere you look" [64].

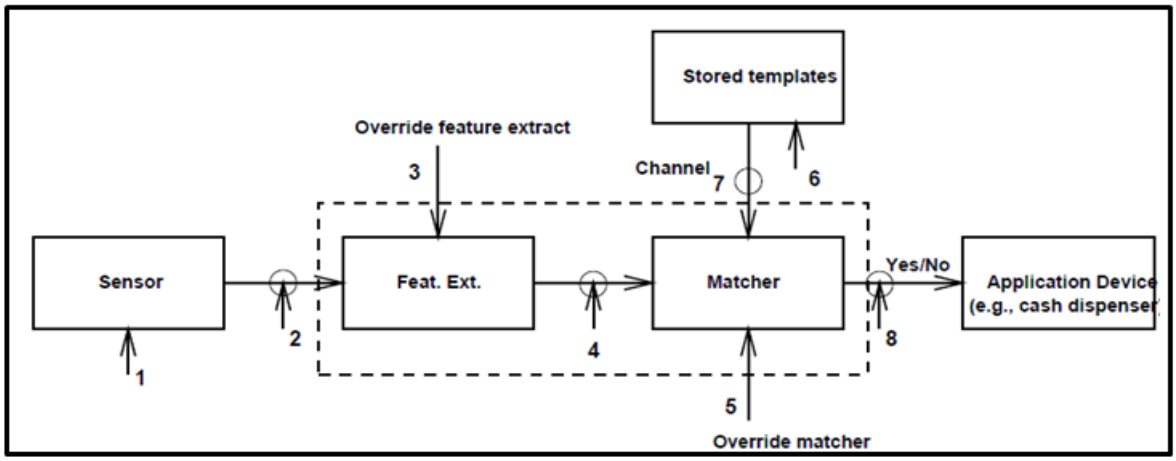


Figure 2.5: Possible attack points in a biometric authentication system [64]

Ratha *et al.* identified several different levels of attacks that can be launched against a biometric system [64]. There are in total eight vulnerabilities or attack points in a biometric authentication system, as illustrated in Figure 2.2 and explained below. Note that each of the vulnerabilities discussed below corresponds to a specific attack point as numbered in Figure 2.2.

1. Fake biometric at the sensor: In this mode of attack, a reproduction of the biometric being used is presented to the system. Examples include a fake finger, a copy of a signature, or a face mask.

2. Resubmission of an old, digitally stored biometric signal: In this mode of attack, an old recorded signal is replayed into the system, thereby bypassing the sensor.

Examples include presentation of an old copy of fingerprint image or recorded audio signal of a speaker.

3. Override feature extractor: The feature extractor is manipulated in such a manner that it produces feature sets chosen by the attacker.

4. Tampering with the feature representation: After the features have been extracted from the input signal, they are replaced with a different synthesized feature set. Often the two stages of feature extraction and matching are inseparable, and this mode of attack is extremely difficult to carry out. However, if the extracted minutiae are transmitted to a remote matcher, then this attack poses a risk.

5. Override matcher: The matcher is attacked to always directly produce an artificially high or low match score.

6. Tampering with stored templates: The database of enrolled templates is available locally or remotely. The attacker tries to modify one or more templates in the database, which could result in authorization for a fraudulent individual, or at least a denial of service for the person associated with the corrupted template.

7. Channel attack between stored templates and the matcher: The templates from the stored database are sent to the matcher through a channel which could be attacked to change the contents of the templates before they reach the matcher.

8. Decision override: If the final result can be replaced by a result chosen by the attacker, the biometric system is rendered useless.

When designing FAS, the vulnerabilities listed above should be carefully taken into consideration.

2.7 Summary

This chapter studied the existing POS transaction processing framework and its system security. The current approaches to mitigate card cloning and their flaws were also studied. In order to evaluate whether biometrics based FAS is a suitable candidate for countering card cloning, a feasibility study was carried out. Even though biometric security offered in the form of an FAS positively confirms an individual's identity and is very popular, it is also susceptible to inherent vulnerabilities associated with biometric authentication systems in general. The next chapter is a continuation of the previous section and analyses various approaches in its aim to propose a robust FAS to address the research problem.

3
Towards a robust FAS

The aim of this chapter is to study the FAS in detail and is structured as follows. Section 3.1 identifies the core entities in a typical FAS, whereas the security and privacy concerns of fingerprint authentication are considered in section 3.2. Section 3.3 recognises and explains the salient features of an ideal FAS. Section 3.4 studies existing solutions that address the vulnerabilities associated with the FASs. The chapter is concluded in section 3.5.

3.1 FAS entities

There are five major entities in a generic FAS, as depicted in Figure 3.1. They are the *sensor*, *feature extractor*, *template database*, *matcher*, and *decision module* [65, 66]. The sensor is responsible for capturing the digital image of a fingerprint pattern [67]. The feature extraction module processes the captured image and extracts its core information, which is referred to as the *feature vector*. The feature vector is used in the subsequent phases of the authentication process. In the fingerprint enrollment phase, the extracted finger template (denoted by X_T) is stored in the system database. Note that the storage medium is dependent on the underlying security model and is not limited to magnetic stripe cards or smart cards. In Figure 3.1, the storage medium used is the system database. The matcher module in majority of the cases is an application program. It receives two biometric feature sets X_T and X_Q (from template and query, respectively) as inputs, and outputs a match score (S). The match score is an indication of the similarity between the two feature sets. In the final phase, the decision module either declares a match or a non-match between the templates, based on the match score [68].

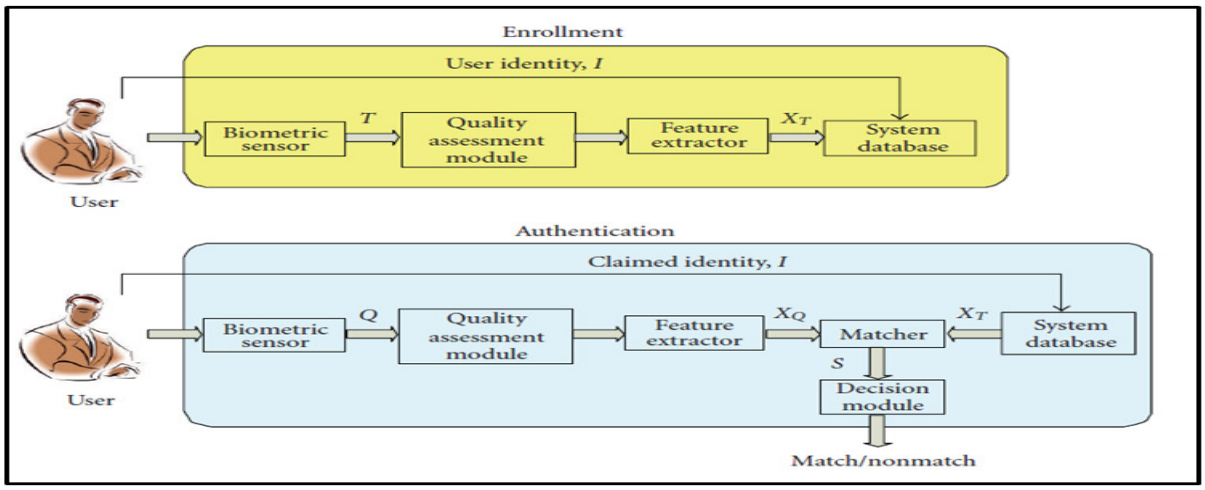


Figure 3.1: A generic FAS [68]

The next section discusses the privacy and security concerns of an FAS.

3.2 Privacy and security concerns of an FAS

The privacy and security concerns of a generic FAS are discussed in what follows [69, 70]. It is noteworthy that these security concerns are also applicable to biometric modalities other than fingerprints.

1. Conventional authentication mechanisms using PINs, passwords, and cryptographic keys have a secret nature and are known only to the user. Fingerprints are authentic, but not disclosed and can be easily recorded, replayed, and abused without users' consent [68, 71]. There are numerous instances where artificial fingerprints such as gelatin fingers have been used to circumvent security systems that rely on fingerprint matching [72, 73]. In contrast to fingerprints, tokens and knowledge have to be willingly shared by the user to be compromised.
2. Fingerprints cannot be revoked or canceled. Passwords and PINs could be changed if compromised, whereas fingerprints are permanently associated with an individual and cannot be revoked if compromised [74]. Although it is possible to enroll multiple fingerprints, there is still a limited choice of fingers to choose from for each individual.
3. If the template is compromised once, its viability as a means of authentication is permanently lost. Fingerprint authentication has advantages in terms of its usability as it obviates the need to remember and manage multiple passwords [75]. However, this also implies that if a fingerprint template is compromised in one application, it is also compromised in other applications where that finger template was used.

4. Cross-matching is used to trace individuals without their consent [76]. Since the finger template might be used in various applications and locations, the user can potentially be tracked if organisations collude and share their respective biometric databases. With traditional authentication schemes, the user can maintain different passwords or credentials to prevent this. The fact that a biometric remains the same presents a privacy concern [76]. The above security and privacy concerns call for various finger template protection schemes which are explained in the next section.

3.3 Finger template protection schemes

According to a study conducted by Christian Rathgeb and Andreas Uhl in their journal "A survey on biometric cryptosystems and cancelable biometrics", an ideal fingerprint authentication system should satisfy four key requirements [70]. They are diversity, revocability, security, and performance, as discussed below.

- **Diversity**: To ensure user's privacy, the underlying scheme should not allow the cross-matching of captured finger templates against multiple databases and applications [70]. This implies that if a user's finger template stored in *database1* is compromised through *application1;* then, *application2* does not allow the matching of the compromised template against the finger template stored in *database2*. This leads to the requirement of being able to generate different templates from the same captured template. In this way, even if one of the systems is compromised, the other systems would stay secure.
- **Revocability:** The scheme should make sure that it is fairly simple to revoke a compromised template and reissue a new one based on the initial template [78].
- **Security:** It must be computationally hard to obtain the original finger template or any template that is close enough to cause a false match from the captured template. This property ensures that it is impossible to reverse engineer a physical copy of the template trait from a stolen template [70].
- **Performance:** The underlying scheme should not degrade the identification and verification performance of the system [78].

The existing finger template protection schemes can be broadly classified under two categories, namely, *cancelable biometrics* and *biometric cryptosystems* [77, 79, 80]. This is illustrated in Figure 3.2.

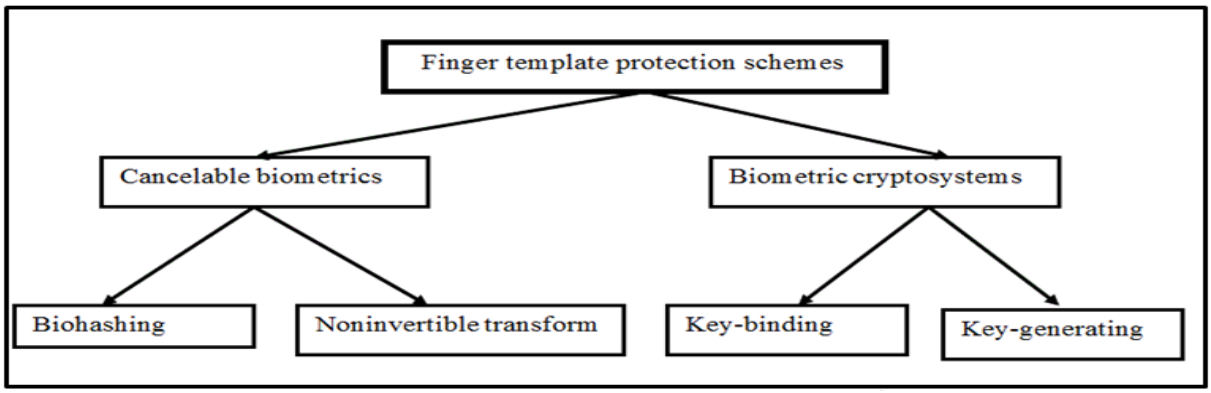


Figure 3.2: Classification of template protection schemes [73]

3.3.1 Cancelable biometrics

Ratha *et al*. introduced the concept of cancelable biometrics and this approach is also known as feature transformation [70]. It consists of a deliberate and iterative distortion of a template based on the selected transformation function. The template is transformed in the same mode at each presentation, for every enrollment and authentication. In this scheme, if the transformed template is compromised, then the transformation function can simply be modified to create a new cancelable template. As illustrated in Figure 3.3, in this scheme, a transformation function f is applied to the template T and only the transformed template $f(T;K)$ is kept in the template database. The attributes necessary for the transformation function are typically derived from a random key K or *password*. The transformation function is then applied to query features Q and the transformed query $f(Q;K)$ is directly matched against the transformed template $f(T;K)$ [68, 69].

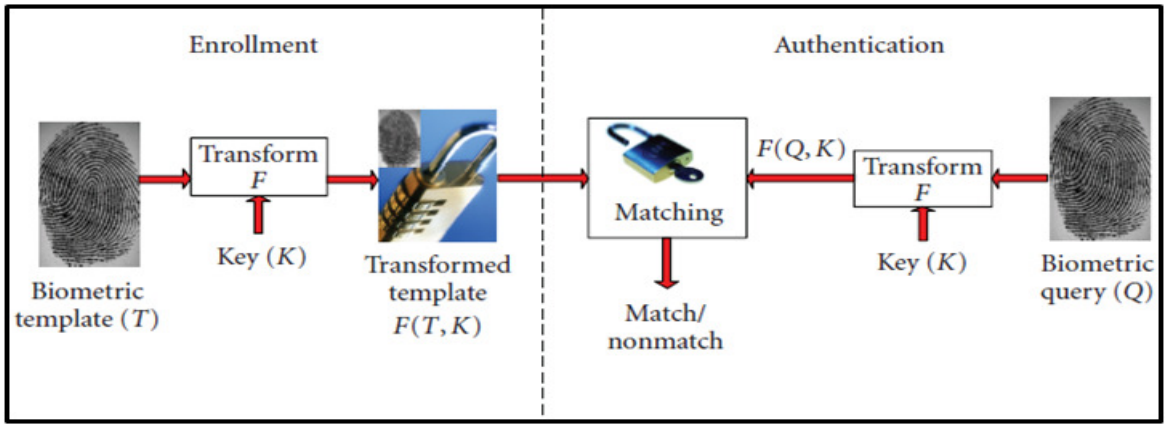


Figure 3.3: Template protection using feature transformation [68]

The cancelable biometric scheme can be further classified into Biohashing and noninvertible transforms based upon the characteristics of the transformation function f.

3.3.1.1 Biohashing

Biohashing is also known as *salting*. It is similar to password "salting" in conventional crypto-systems. In this approach, before hashing, the *password P* of the user is concatenated with a pseudorandom *string or number S* and the resulting *hash H(P+S)* is stored in the database [65]. The biohashing is based on the same principle.

Biohashing was introduced as a form of cancellable or replaceable biometrics, in which a set of user-specific random numbers are integrated with biometric features to address the problem of privacy and security [68, 81, 82]. In this scheme, the template transformation is based on a function defined by a key or a password.

Biohashing schemes have been proposed for iris and palmprint modalities [83, 84]. Another example of biohashing is the cancelable face filter approach proposed for face recognition. In this scheme, a set of user-specific random attributes are convolved with the face images during enrollment and authentication [85].

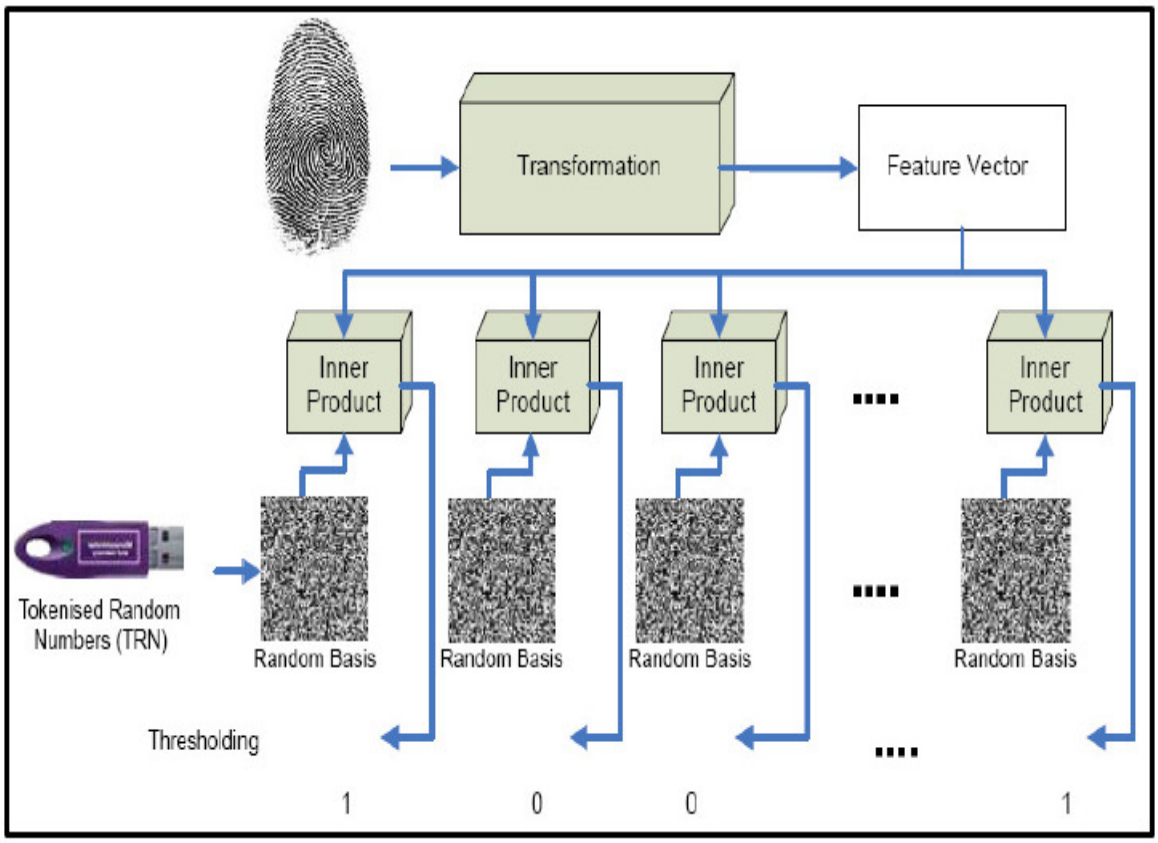


Figure 3.4: Template protection using Biohashing [88]

The progression of biohashing is shown above in Figure 3.4. Initially the fingerprint undergoes a transformation and the features in the fingerprint are extracted. These extracted features are called feature vectors. The feature vectors contain various features of the provided fingerprint, which are used for generating the hash code. The system then receives the Tokenised Random Number (TRN) as an input. The next step is to combine the supplied TRN with the feature vectors obtained from the fingerprint. The inner product resulted from the combination is essentially the biometric hash code generated for the provided fingerprint and the TRN [88].

The transformation function used in biohashing is invertible to a large extent, provided the key or password that is used for biohashing is compromised [65]. Inversion is the process of recovering the original biometric template from the transformed template and invertibility can be expressed in terms of the computational complexity and the number of guesses involved in recovering the original template [86].

In the biohashing approach, it is a prerequisite that the key or the password needs to be securely stored and recollected by the user, and presented during authentication [76]. The key is used to increase the entropy of the template and thus makes it difficult for the adversary to guess the template. The entropy of a biometric trait is a measure of the number of different identities that are distinguishable by the biometric system [76].

The usage of the same template data in biohashing can result in highly correlated bit strings because of the high tolerance of data capture offsets [68]. Moreover in this approach, there is no deterministic way to get the user specific code without having both the key and the template. Thus, biohashing could protect the biometric system against any biometric fabrication and therefore the imposters can be easily identified.

Biohashing facilitates the revocation of the template in case if it is compromised. It also prevents cross-matching between the databases, since each application using the same biometric uses a different transformation [65]. Furthermore, biohashing offers significant functional advantages such as zero error rate point and clean separation of the genuine and imposter populations, thereby allowing FAR elimination without suffering from increased occurrence of FRR [68].

The next subsection will explain the second class of the cancelable biometric scheme, which is the noninvertible transform.

3.3.1.2 Noninvertible transform

In this scheme, the template security is based on the noninvertible transformation function. A noninvertible transform can be understood as a one-way function, f, that is "easy to compute" but "hard to invert" [68]. In other words, if $f(x)$ is known, the probability of deriving x in polynomial time is small [80]. The main feature of this scheme is that even if the key or the transformed template is known; it is computationally hard for an adversary to discover the original template. Apart from cancelable biometrics, the other finger template protection scheme is biometric cryptosystems, which is explained in the next section.

3.3.2 Biometric cryptosystems

Biometric cryptosystems were initially developed for the purpose of either securing or generating a cryptographic key to or from templates [68, 76]. Nevertheless, they can also be used as a template protection mechanism. In this scheme, some public information about the template known as *helper data* is stored. Although the helper data does not reveal any significant information about the original template, it is necessary during matching to extract a cryptographic key from the query templates. Matching is performed indirectly by verifying the validity of the extracted key [76]. In this scheme, error correction coding techniques are used to handle intra user variations. This scheme is illustrated in Figure 3.4.

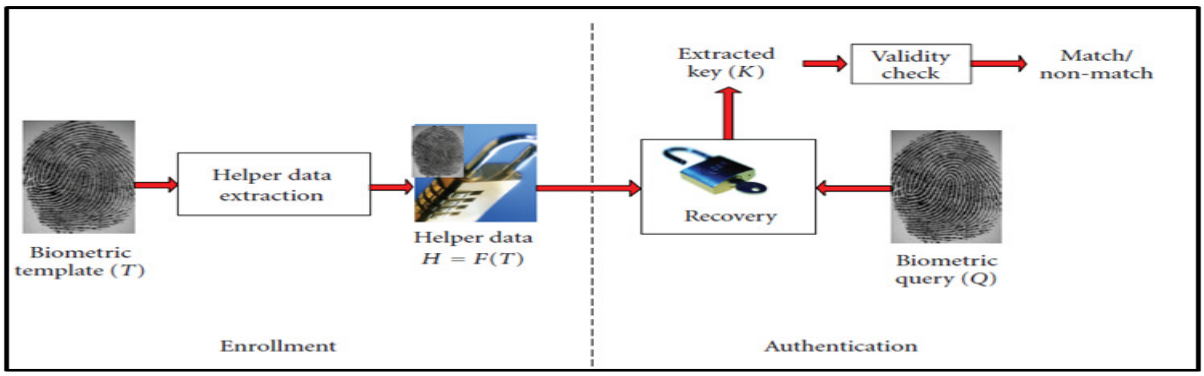


Figure 3.5: Template protection using biometric cryptosystem [68]

Biometric cryptosystems can be further classified into *key-binding* and *key-generating* schemes depending on how the helper data is obtained, and they are discussed below [68].

3.3.2.1 Key-binding scheme

In this scheme, the template is captured by monolithically binding it with a key within a cryptographic framework [81]. Here, the helper data is stored in the database, which is basically a combination of the template and the key. It does not reveal much information about the template or the key, and it is computationally hard to decode the template or the key without any knowledge of the user's template.

3.3.2.2 Key-generating scheme

In this scheme, the helper data is derived directly from the template, and the cryptographic key is directly generated from the helper data and the query biometric features [81]. A key *K(T)*, parameterized by the template *T*, is stored instead of the actual biometric itself. During verification, it is checked if *K(T') = K(T),* where *T* corresponds to the enrolled template and *T'* corresponds to the query template [81]. The generation of the keys directly from the templates is an appealing template protection approach which can also be very useful in cryptographic applications.

The next section will methodologically analyse the existing template protection schemes by explaining their advantages and disadvantages.

3.4 Analysis of the current template protection schemes

This section describes the security aspects of the current template protection schemes in a holistic and systematic manner [80]. It is summarised as in Table 3.1.

Table 3.1: Analysis of different template protection schemes

Template protection schemes	Pros	Cons
Noninvertible transform	1. Offers high security 2. Diversity 3. Revocability	It is nearly impossible to design a transformation function that satisfies both discriminability and noninvertibility simultaneously.
Biohashing	1. Low FAR 2. Diversity 3. Revocability	The template is no longer secure if the key is compromised.
Key-binding	Tolerant to	1. Reduction in matching accuracy.

	intra user variations in the template.	2. The helper data needs to be chosen very carefully.
Key-generating	Highly beneficial in cryptographic applications.	Extremely difficult to generate keys with high entropy and stability and hence practical implementations are not feasible.

As explained in section 3.3.1, cancelable biometrics use transformed or intentionally-distorted biometric data instead of raw data for authentication. The use of this scheme is being increasingly applied to address the security vulnerabilities specific to biometric authentication systems. Although this approach bridges the gap between the convenience of biometric authentication and security vulnerabilities, there is a risk in using this scheme. The transformation of the templates will decrease the performance of the underlying authentication system [76]. This is attributed to the fact that the complexity of the transformed biometric is much higher than that of the original data. Ratha *et al*. introduced *cancelable biometrics* to alleviate the problem of compromised templates [87]. Although randomisation was proposed as a general way of producing cancelable templates, they did not provide any specific and practical functions to prove it.

As outlined in Table 3.1, noninvertible transforms offer high security, diversity, and revocability. The main potential drawback of this scheme is the implicit trade-off between the discriminability and noninvertibility of the transformation function [65, 76]. The transformation function should preserve the discriminability or the similarity structures of the feature set. It implies that just as in the original feature space, features from the same user should have high similarity in the transformed space. Further, the features from different users should be relatively dissimilar after the transformation. The transformation should also be noninvertible, which implies that, given a transformed feature set, it should be hard for an adversary to obtain the original feature set or a close approximation of it.

Biohashing protects biometric privacy with an external confidential key or a tokenized pseudo random number [65]. The usage of the key in the scheme offers low FAR, diversity, and revocability. However, the security of this scheme is dependent on the

security of the keys or tokens and they are easy to be lost, stolen or compromised [65]. In biometrics, multiple acquisitions of the same template do not necessarily result in the same feature set and hence one cannot store a template in an encrypted form and then perform matching in the encrypted domain. As a result of this, biometric templates cannot be stored and matched in the encrypted domain. Moreover, encryption is typically not a smooth function and a small difference in the values of the feature sets extracted from the raw biometric data would lead to a very large difference in the resulting encrypted features [65].

There is no guarantee that the matching between an enrolled and a query template in the encrypted state will be consistent because of the differences of biometric data and the general nature of cryptographic functions, which produces totally different results for analogous inputs. Although it is possible to decrypt the query template and then perform the matching of the original template, such an approach is not secure. This is because a Trojan horse type attack can reveal the template and expose it during every authentication attempt [73]. Furthermore, the standard encryption algorithms do not support a comparison of the templates in the encrypted domain. Thus, templates are always exposed during every authentication attempt. The use of biometric cryptosystems leads to a significant decrease in recognition performance. This is due to the fact that, the enrolled template is not available within the cryptosystem, and therefore cannot be aligned properly during the comparison stage. Hence, standard encryption techniques are not beneficial and are not appropriate for securing biometric templates.

As laid out in Table 3.1, the key-binding scheme is tolerant to intra-user variations on the template, and is dependent on the error correcting capability of the accompanying code word. In this scheme, it is a prerequisite that the matching is always carried out using error correction schemes. The disadvantage is that matchers known to be accurate cannot be deployed in this scheme, which can possibly lead to a reduction in the matching accuracy. Another drawback of the scheme is that the helper data need to be carefully derived and should be based on the specific biometric characteristics.

Although the key-generating scheme is highly appealing and useful in cryptographic applications, it has the following disadvantages. It usually suffers from low discriminability which can be measured in terms of *key stability* and *key entropy*. Key

stability refers to the extent to which the generated key from the template is repeatable, and key entropy corresponds to the total number of potential keys that can be derived. If the scheme generates the same key irrespective of the input template, it has high key stability, but zero entropy, which implies a high FAR. On the other hand, if the scheme generates distinctive keys for different templates of the same user, the scheme has high entropy, but no stability and this leads to high FRR. Although it is possible to generate a key directly from the template, it is difficult to achieve high key entropy and key stability at the same time. Moreover, practical implementation of this scheme has not been demonstrated yet [81].

The integrity and security of templates can be guaranteed only to a limited extent by the existing methods. As a result, their usage can open doors to severe security vulnerabilities [87]. The major challenge in designing a finger template protection scheme that satisfies all the requirements (addressed at the beginning of section 3.3) is the need to handle intra user variability in the acquired templates. It is vital for an authentication scheme to have a good trade-off between accuracy and security. A finger template protection scheme with robust security and acceptable recognition performance, at the present time has remained unclear. The development of such a system is highly essential as biometric systems are beginning to flourish into the core physical and information infrastructure of our society. The next section summarises the current chapter.

3.5 Summary

This chapter analysed the various aspects of a FAS in its aim to address the research problem. It first identified and elaborated the key entities of an FAS from an enrollment and authentication perspective. Furthermore, the security and privacy concerns that are specific to the FAS were addressed. This was followed by the classification of finger template protection schemes. The key finger template protection schemes proposed by various researchers were studied. It was found that existing template protection schemes are either vulnerable, or have performance issues. In addition to this, some of the schemes were not stable and lacked practical implementations. Thus, it was concluded in this chapter that the usage of existing schemes can lead to severe security vulnerabilities, and that there is a need to propose a robust FAS. The next chapter proposes a candidate FAS.

4
Proposed FAS

4.1 Proposed FAS (PFAS)

The application scenario and its requirements play a major role in determining the characteristics of an appropriate FAS. In addition, it is also important to consider a number of other factors such as privacy, memory requirements, user acceptance, recognition performance, computational complexity, and cooperation. Apart from these factors, it is critical that the proposed system should solve the problem addressed by the research. From the analysis and study of Chapter 3, it is concluded that a derivative of the Biohashing scheme will be the most appropriate for solving the current research problem. In addition, this derivative addresses a subcategory of the vulnerabilities associated with the FASs. It should be noted that this research does not solve all the inherent vulnerabilities associated with the FASs that were identified in section 2.6 of Chapter 2; instead, the focus is on solving a specific subset of the fingerprint security vulnerabilities that are relevant to the current research context. They are formulated in the form of objectives of the proposed FAS and are presented in section 4.1.1. The proposed FAS from here on will be known as PFAS.

4.1.1 Objectives of the PFAS

The key objectives that the PFAS is required to achieve in addressing the research problem are as follows:

- Preserve the privacy and security of the template data.
- Mutual authentication.
- Be resilient to the compromise of the template itself.
- Support revocation of the template, in case it gets compromised.
- Be resilient to replay attacks so that the replay of the template does not result in a successful authentication.

The security model for the PFAS will be designed based on these key objectives, and is explained in the next section.

4.1.2 The proposed security model

To solve the research problem and to achieve the objectives of the PFAS, this research proposes a security model that is based on the principles of Biohashing and One-Time Password (OTP) scheme. The key concept behind Biohashing is to apply a transformation function on a biometric feature, thereby transforming it into another domain [88]. The OTP scheme is a well-known authentication technique in electronic transactions, where an OTP is generated for authenticating a particular transaction session [89]. The password generated can be used only once and hence robust security can be achieved. Therefore, the proposal is to derive an unpredictable, one-time finger template by inheriting the principles of Biohashing and OTP scheme. The *One-Time Template* (OTT) will be generated based on those parameters that are pertinent to the current transaction session. During the authentication phase, the OTT derived for authenticating the user is matched against the OTT at the authentication server, and the decision is made as to the authenticity of the user.

The process flow of the security model is presented as a sequence of steps in what follows.

Step 1: During the enrollment phase, the finger template of the user, along with a set of key user attributes is registered in the Biometric Authentication Server (BAS). A sample attribute list is provided below. Note that attribute lists in general are not limited to the following entities, and that other user-specific attribute entities can be included as well.

<attribute 1> : *<ID >*
<attribute 2> : *<SSN>*
<attribute 3> : *<PIN>*
<attribute 4> : *<password>*
...
<attribute n-1> : *< email id >*
<attribute n> : *<mother's maiden name>*

In the current security model, the attribute list is limited to 8 entries as it is sufficient to generate the expected level of security, and at the same time it saves the average time to enroll each user. The enrollment and card issue process are illustrated in Figure 4.1.

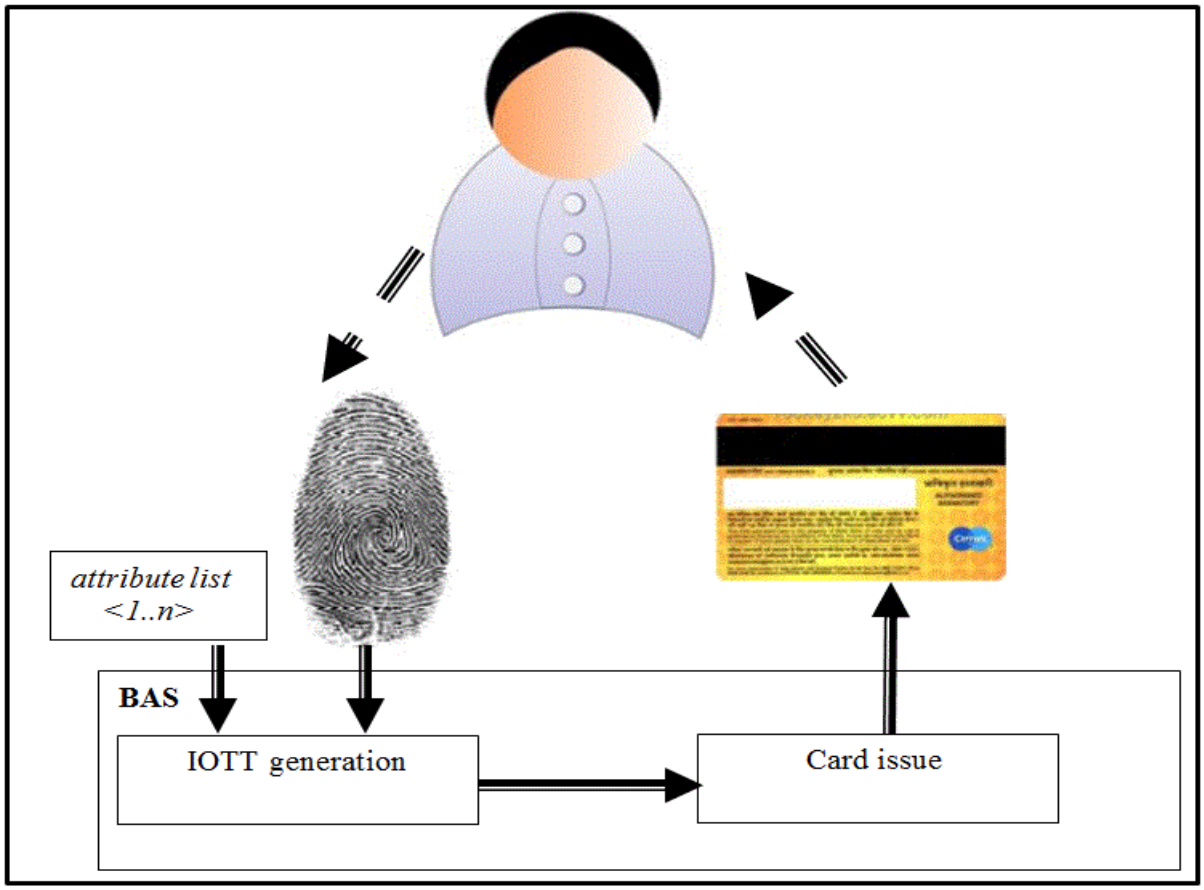


Figure 4.1: Enrollment and card issue process

Step 2: The BAS is equipped with an intermediate OTT generation module. This module takes as input the extracted finger template and the attribute list. The user's *Intermediate One-Time Templates* (IOTTs) are subsequently generated for each *Transaction Number* (TSN), based on the *Biometric Keys* (BKs) derived from the attribute list captured in Step 1. The enrollment and card issue process is complete at this stage. The IOTT generation process is illustrated in Figure 4.2.

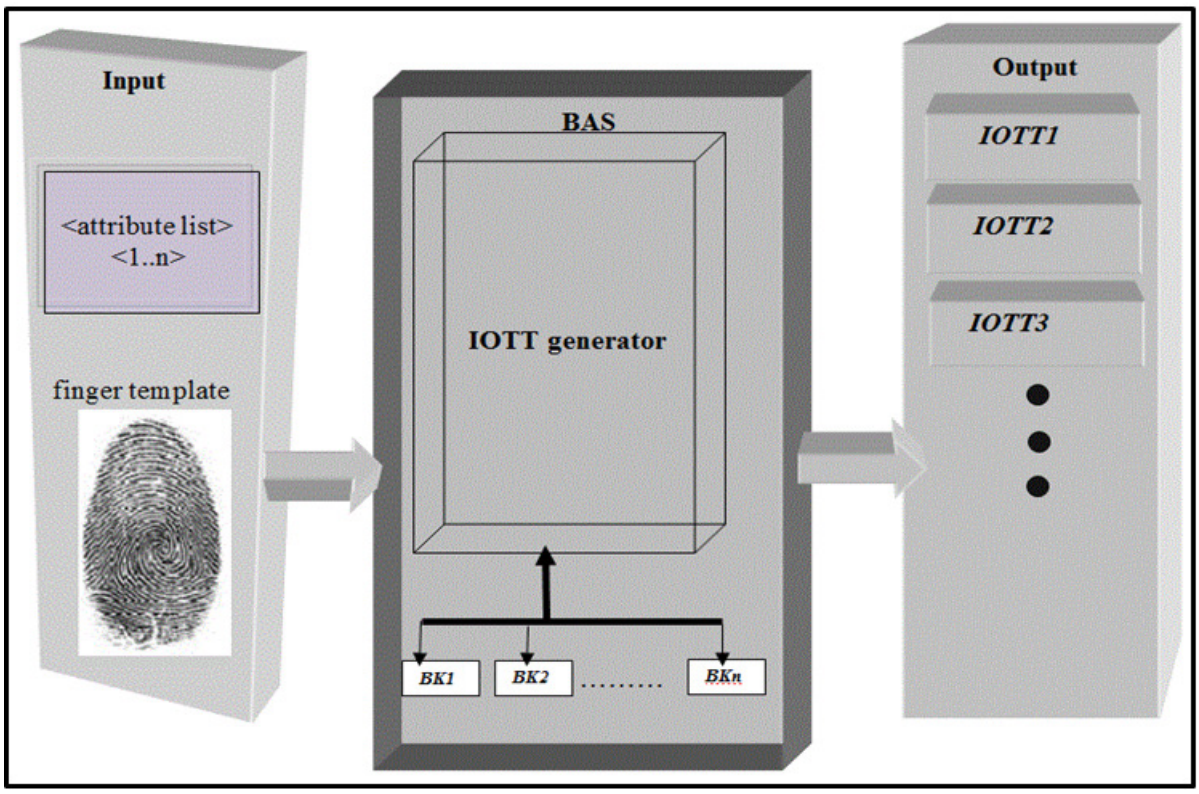


Figure 4.2: IOTT generation

The TSN, BK and the corresponding IOTT list are mapped to the user PAN in the IOTT database, as illustrated in Table 4.1., where xxxxxxxxxxxxxxxx is the 16 digit PAN corresponding to the user.

Table 4.1: IOTT database

PAN: xxxxxxxxxxxxxxxx		
TSN	**BK**	**IOTT**
TSN1	*BK1*	*IOTT1*
TSN2	*BK2*	*IOTT2*
…	…	…
TSNn	*BKn*	*IOTTn*

The value of *n* in Table 4.1 is limited to 73200, which this study considers as ample from the following hypothesis. If a user performs 50 transactions per day, and continuously throughout the year (taking leap years into consideration) for a period of 4 years, then the user will be performing a maximum of 73200 transactions, which is considered to be more than is expected. A general bussiness rule and norm in the banking industry is to replace bank cards after 3 or 4 years, or after the specific expiry date [34]. This is essentially

because of card limitations such as maximum transactions that can be performed during the lifetime of a card, wear and tear of the card, and to mitigate card fraud [34]. Note that in this approach, only the transformed IOTTs are stored and the original user templates are never stored in the BAS.

Step 3: On the client side, the magnetic card swipe in the POS triggers transaction processing. The POS then captures the PAN, the current date stamp, and the time stamp. These fields are subsequently encapsulated in an online logon request message, which is sent to the BAS.

Step 4: The BAS on receiving the logon request, queries its user template database against the PAN. The appropriate BK that is pertinent to the transaction and the TSN is retrieved. They are then assembled in a logon reply message, and transmitted to the client. This is illustrated in Figure 4.3. Note that the TSN gets incremented after every authentication attempt from the client.

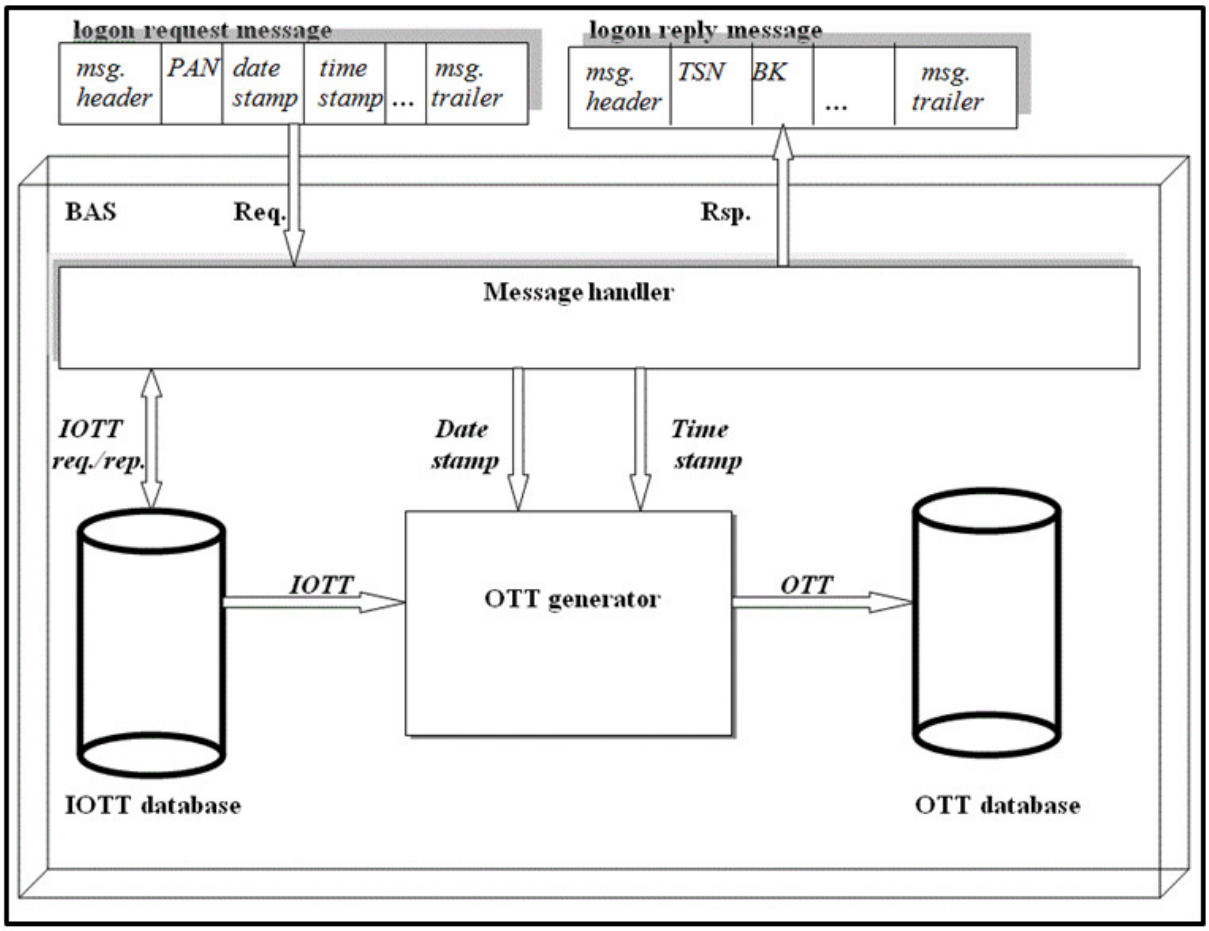


Figure 4.3: OTT generation in the BAS

Step 5: After sending the logon reply, the BAS generates an OTT, which is used for authenticating the current transaction session. If this were the first transaction attempt for the client, it implies that *TSN1* and *BK1* were used in deriving *IOTT1*. The *IOTT1* is subsequently transformed using the current date stamp and time stamp received in the logon request to derive the actual OTT. In this case, *OTT1* will be generated. The role of the current date stamp and time stamp as extra security salts in the protocol is to strengthen the security of the generated OTT, and to add an element of unpredictability in future transactions. The OTT generation process in the BAS is illustrated in Figure 4.3.

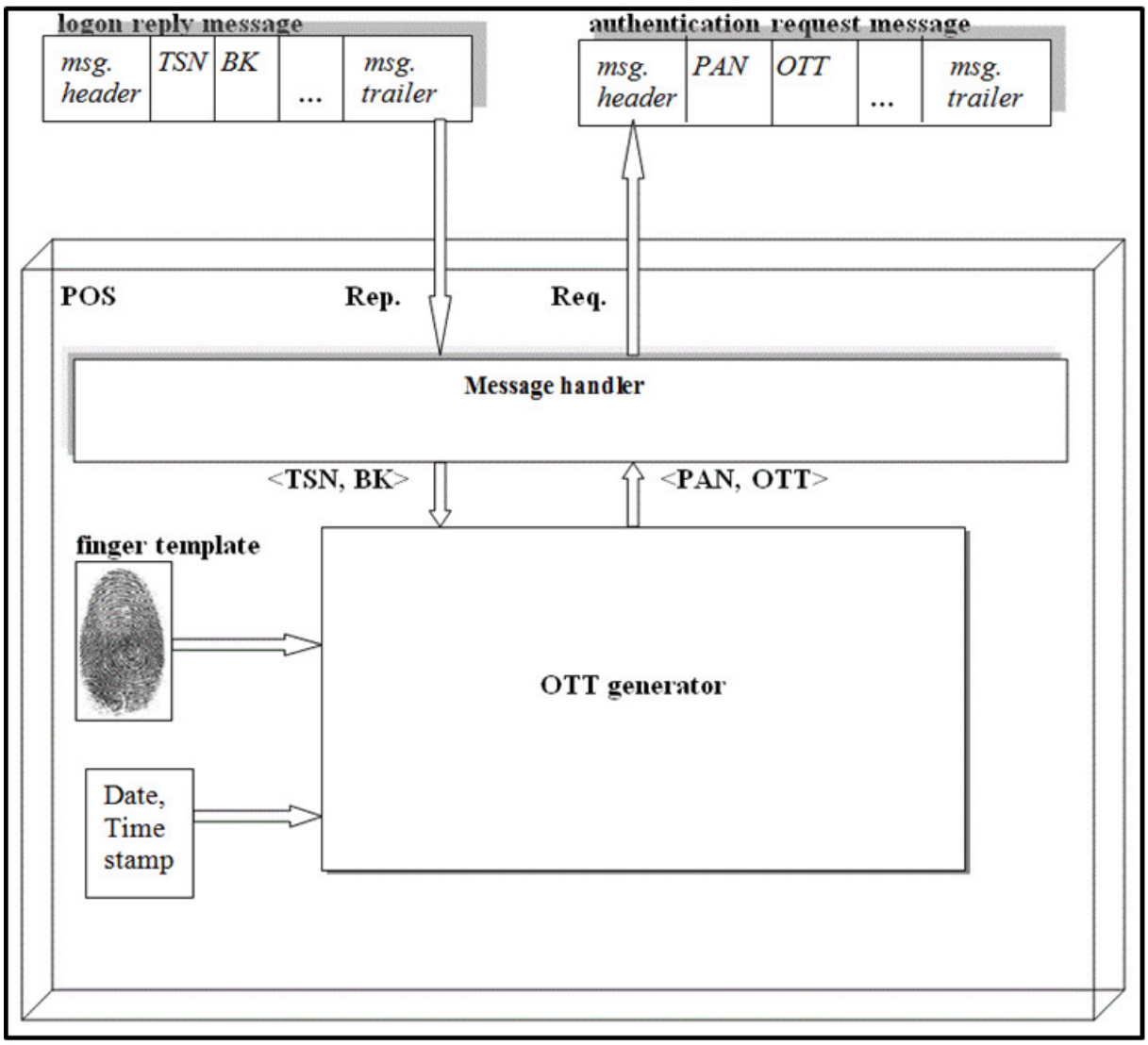


Figure 4.4: OTT generation in the POS

Step 6: The POS, on receiving the logon reply, captures the finger template of the user. It then follows the transformation process based on the *TSN* and the *BK* that was received from the BAS in the logon reply. The IOTT will be generated and transformed using the current date stamp, and the time stamp to generate the actual OTT. In this case, *OTT1* will be derived from *IOTT1* and is sent as an online authentication request to the BAS. This process is illustrated as in Figure 4.4.

Step 7: The BAS, on receiving the authentication message compares the OTT generated from the POS against the OTT generated at the BAS and decides if the OTTs match, based on the decision threshold set in the BAS. The status is sent down to the client as an

authentication reply. After sending the authentication reply, the BAS deletes the OTT for that transaction session.

Step 8: The POS on receiving the authentication reply, checks the response code. If the response code is a success, the POS proceeds with the rest of the transaction processing as normal. If the response code is a failure, then the transaction processing is terminated at this stage. The authentication message sequence between the POS and the BAS are depicted in Figure 4.5.

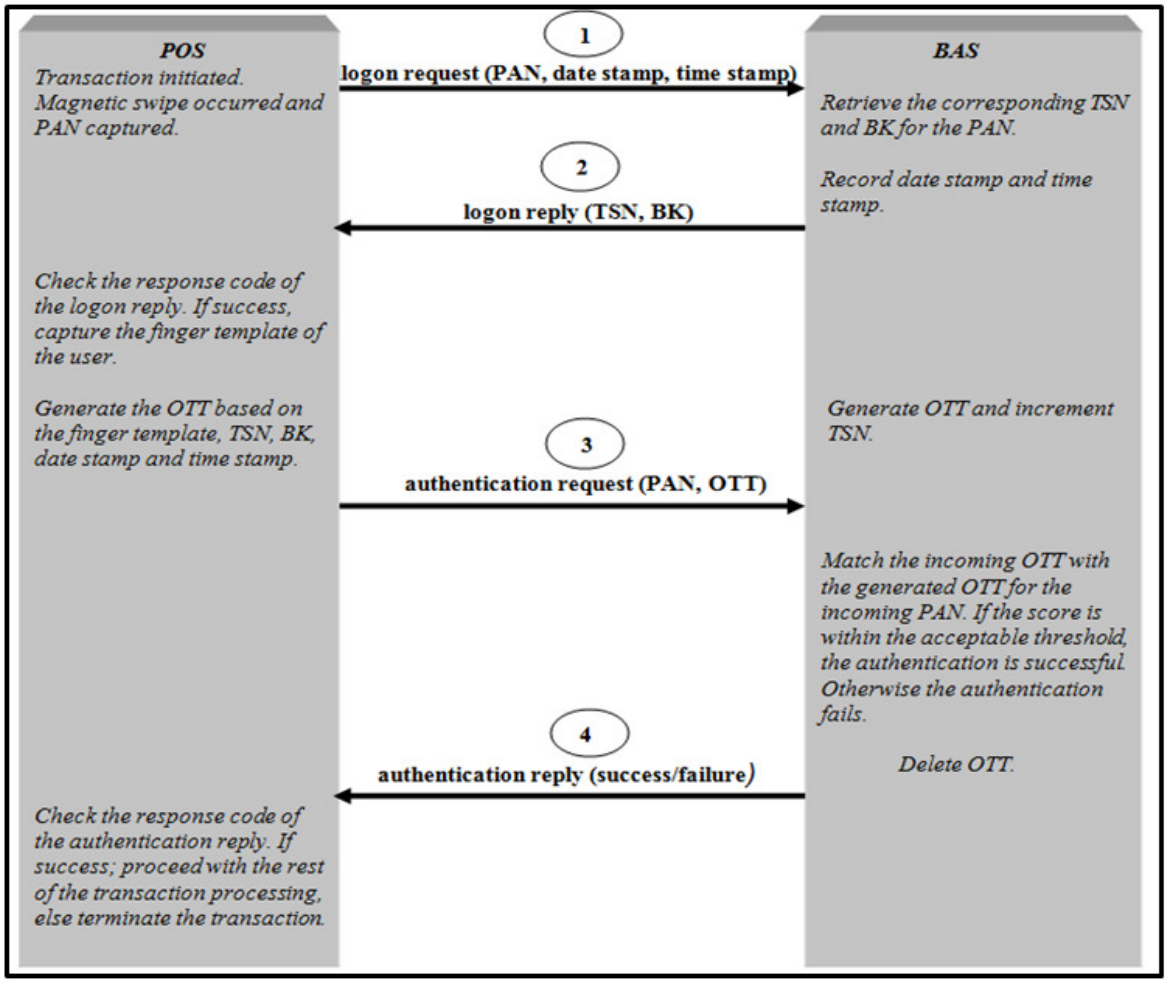


Figure 4.5: Authentication message sequence

The communication algorithm holding the security model is summarised as in Figure 4.6.

1. During the enrolment phase, the attribute list of the user is captured along with the finger template.
2. Each IOTT template is generated for the user based on the IOTT algorithm as illustrated in Figure 4.7 The IOTT generated thus are mapped against the corresponding TSN and BK under the user's PAN. The original finger template is never stored at the BAS. At this stage, the enrolment is complete.
3. During the transaction phase, the user's magnetic stripe card is swiped at the POS. The PAN is captured; the date stamp and time stamp are recorded as well. A logon request message is then assembled and is sent to the BAS.
4. The BAS on receiving the logon request parses the message and retrieves the user's PAN, date stamp and the time stamp.
5. The BAS retrieves the TSN and the BK for the corresponding PAN. It then assembles the TSN and the BK for the current transaction session in the logon reply and sends it to the POS.
6. In the BAS, the OTT for that authentication session is generated from the corresponding IOTT.
7. The POS on receiving the logon reply extracts the TSN and the BK. It then generates the IOTT based on the OTT algorithm illustrated in Figure 4.7.
8. The OTT will be generated from the corresponding IOTT.
9. An authentication request message is then assembled with the PAN and the OTT is sent to the BAS.
10. The BAS on receiving the authentication request disassembles the PAN and the OTT. The OTT from the POS is then verified against the OTT generated at the BAS.
11. The transaction is authenticated or declined based on the decision threshold. The authentication result is populated in the authentication reply message and is sent to the POS.
12. The POS on receiving the authentication reply continues with the rest of the transaction processing based on the authentication result received from the BAS.

Figure 4.6: Algorithm for the communication between the POS and the BAS

The OTT is generated based on the OTT algorithm, which is explained in the next section.

4.1.2.1 OTT algorithm

The algorithm for generating the OTT, as illustrated in Figure 4.7, is presented in what follows. Note that for compatibility between the POS and the BAS, it is mandatory to implement the same OTT algorithm on both sides.

Compute OTT

Input :

finger template, attribute list: a1..a8

Output :

OTT

1. ***fv ⟵extract_feature_vector(ft)***

2. ***w1..w8 ⟵map_attributes_to_weights(a1..a8)***

3. ***BK1..BKn ⟵derive_BioKeys(w1..w8)***

where n *is* 128!

4. ***for each*** *BK from 1 to n*

TTk ⟵transform_BioTemplate(fv)

where k *is the* k*th element*

5. ***for each*** *TSN from 1 to n*

IOTTi ⟵generate_IOTT(TTi)

where i *is the* i*th element*

6. ***OTTcs ⟵create_OTT(DS, TS, IOTT)***

*where OTT*cs *implies the OTT generated for the current session*

Figure 4.7: Algorithm for generating OTT

As illustrated in Figure 4.1, during the enrollment phase, the finger template and the attribute list are acquired from the user. The finger template is denoted by *ft* and is presented as input to the ***extract_feature_vector*** routine to generate the corresponding feature vector (denoted by *fv*). This routine extracts the raw minutiae from the finger template and will store the same in the *fv*. The *fv* will then convert each minutia into a 4-tuple format that defines the minutia. The X co-ordinate, the Y co-ordinate, the

orientation, and the minutia type (whether it is a ridge ending or a ridge bifurcation) are derived for the whole minutia set and are stored respectively in a 4-tuple format in the *fv*.

The next step is to map the attributes captured from the user to weights. This is carried out using the ***map_attributes_to_weights*** routine. Each attribute is first converted to a string of 16 bytes. If the attribute length is shorter than 16 bytes, then it is padded with extra characters. After this step, the attributes *a1..a8* are mapped to the corresponding weights *w1..w8*, where the weights are converted to their corresponding 16 byte numeric representations.

The next step in the algorithm is to generate the *Biometric Keys* (*BKs)* using the ***derive_BioKeys*** routine. In this, a randomisation function accepts the weights from *w1* to *w8* and takes care of the permutation of the *BKs.* Although *BK* can be any one of 128! (128 factorial) combinations as each weight is a 16 byte field; it is a very large number and will have implications on the storage availability in the real world. The *BK* range is purely dependent on the security requirement and the storage availability of the application that is making use of this algorithm. For the current study, the maximum number of *BKs* that can be generated is set to 73200, as elucidated in the previous section.

The next step of the algorithm is to generate the *Transformed Templates (TTs)* and is generated using the ***transform_BioTemplate*** routine. The *TTs* are basically derived using a transformation function that takes as input *BK* and *fv* obtained in steps 1 and 3. The *TT* corresponding to each *BK* is generated in this routine. It is represented as follows in Figure 4.8.

$$T(fv, BK) \rightarrow TT \qquad (1)$$

In equation (1), *T* is the transformation function that takes as input *fv* and *BK* and transforms it into *TT*. The transformation function is derived by following the geometric transformation approach. This approach is applied in the fingerprint minutiae feature to transform minutiae location and orientation. The *BK* in equation (1) is a 128 byte binary string which is represented in the following format.

BK →

1	0	0	1	1	0	0	1	1	0	0	0	1	0	1	1	1	1	...	0

The *fv* in equation (1) is an *nx4* matrix of the following format, where *n* is the number of minutiae points.

fv →

X	**Y**	**Ø**	**β**
x1	*y1*	*ø1*	*β1*
x2	*y2*	*ø2*	*β2*
.	.	.	.
xn	*yn*	*øn*	*βn*

The X, Y, Ø, and β fields of the matrix represents the X co-ordinate, the Y co-ordinate, the orientation, and the minutiae type. A transformation function is applied to the *fv* for each *BK* to generate the corresponding TT. The transformation function is based on the *XOR* (Exclusive OR) masking of *BK* with each element of the *fv* . The *TT* generated thus will be of the following format.

TT →

X'	**Y'**	**Ø'**	**β'**
x1'	*y1'*	*ø1'*	*β1'*
x2'	*y2'*	*ø2'*	*β2'*
.	.	.	.
xn'	*yn'*	*øn'*	*βn'*

where the X', Y', Ø', and β' represents the X co-ordinate, Y co-ordinate, orientation, and minutiae type of the *TT*.

Figure 4.8: Geometric Transformation function

The next step in the algorithm is to derive the *IOTTs* and is generated using the ***generate_IOTT*** routine. The *IOTTs* are derived by salting the corresponding *TSN* and the corresponding *TT*. During an authentication session, the respective *IOTT* is passed as a parameter to the ***create_OTT*** routine. In this routine, the *Date Stamp (DS)* and *Time Stamp (TS)* that are extracted from the logon request message are applied or salted with the *IOTT* to create the *OTT* for the current authentication session. The implementation of the OTT algorithms at the POS and the BAS are further described in section 7.1 of Chapter 7 and

are also depicted in Table 7.1 and 7.2 in the appendix. The next section focuses on analysing the complexity of the OTT algorithm.

4.1.2.2 Complexity analysis of the OTT algorithm

An estimate of the complexiy of the OTT algorithm is derived in this section. Since an algorithm may have different execution times based on external factors such as processor speed, instruction set, operating system, compiler type, and input size, the OTT algorithm complexity will be measured asymptotically [56]. This implies that the complexity will be calculated as a factor of the number of steps or operations that are needed to execute the algorithm.

In Figure 4.7, there are 6 steps in the OTT algorithm. Let *n* be the number of minutiae in the minutiae set, so the complexity of performing step 1 becomes *4*n* (as each minutia will be defined as a 4 tuple). The complexity of step 2 is *8*, as there are *8* attributes that need to be converted to their respective weights. Step 3 derives the *BKs* from *1* to *n* by taking weights *(w1..w8)* as inputs, where each weight is 16 bytes. Therefore, *n* can go up to a maximum of *n!*. The complexity of step 4 to 6 is *4*n* for each step. From the analysis of the complexity of each step, it is concluded that the algorithm executes with complexity that is linear in the input size. Thus, the computational complexity of the algorithm is estimated to be *O(n),* where *n* is the input size.

Following that this section has explained the PFAS, the next section concludes this chapter.

4.2 Summary

This chapter conceptualised the proposed FAS (PFAS). The objectives of the PFAS were presented, and the security model was designed. This was followed by the explanation of the processes for fingerprint enrollment and the issuing of magnetic stripe bank cards. The OTT generation process and the step-by- step communication sequence between the POS and BAS were also presented. Furthermore, the OTT algorithm was derived and its complexity analysis was conducted. The next chapter analyses the individual building blocks of the PFAS and the underlying security protocol in detail.

5
Detailed analysis of the PFAS

In this chapter, the individual building blocks of the PFAS are analysed. The chapter layout is as follows. Section 5.1 models the key components of the PFAS using a Structured System Analysis and Design Methodology (SSADM). Section 5.2 focuses on the underlying FAS protocol that drives the PFAS, whereas section 5.3 concludes the current chapter.

5.1 An SSADM for the PFAS

This section aims to conceive an SSADM for the PFAS. In the following subsections, the system is designed using the Data Flow Modeling or Data Flow Diagramming (DFD)[3] techniques. The individual DFD models are then assembled using a bottom-up approach to create a complete PFAS data flow model. The reason for selecting the DFD technique is that in this technique, one system can be logically partitioned into subsystems, which can be further partitioned into subsystems at a very low level. Every subsystem in a DFD corresponds to a process in which input data is processed, and the processes cannot be partitioned further after reaching a certain lower level. Each process in the DFD characterises an entire system on its own. In this methodology, data introduced into the system is from an external entity, and once data enter the system, it flows between processes. Each process produces output data that may or may not serve as an input to another process [90]. The individual building blocks in the framework are laid out as follows.

[3] DFDs are used in a software-development system to describe the various components of a software system in terms of its data flows.

5.1.1 Enrollment

The enrollment process is as illustrated in Figure 5.1. This process is carried out at the bank where the user holds his or her transaction account. The labels numbered in the figure represent the sequence of events in the enrollment process. In this process, the attribute list and the finger template are first captured in the *Enrollment Request(1)* event. The *enrollment entity* first extracts the relevant minutiae points corresponding to the finger template. It then encapsulates the attribute list to the extracted minutiae set and sends an *IOTT Generation Request(2)* to the *enrollment/IOTT interface* for further processing. The interface processes this request and sends an *IOTT Generation Response(3)* to the *enrollment entity*. The *enrollment entity* subsequently sends a *Card issue Request(4)* to the *card issue entity*. The *card issue entity* sends a *Card issue Response(5)* and a new magnetic stripe bank card is issued (*New Bank card issue(7)*) to the user. If the IOTT generation is unsuccessful, a negative *IOTT Generation Response(3)* is sent to the *enrollment entity*. In this case, the *enrollment entity* sends an unsuccessful *Enrollment Response(6)* to the user, and it requests that re-enrollment be initiated.

The *enrollment entity* does not store the original finger template and the attribute list of the user. Once the *IOTT Generation Response(3)* is received, the original finger template and the attribute lists that were captured are deleted. *In this manner, the possibility of a replay attack is eliminated during the enrollment phase.* In order to optimize the enrollment time per user, this research assumes that only a single finger template is captured and the attribute list is limited to eight entries.

The IOTT generation and the storage process are explained in detail in the next section.

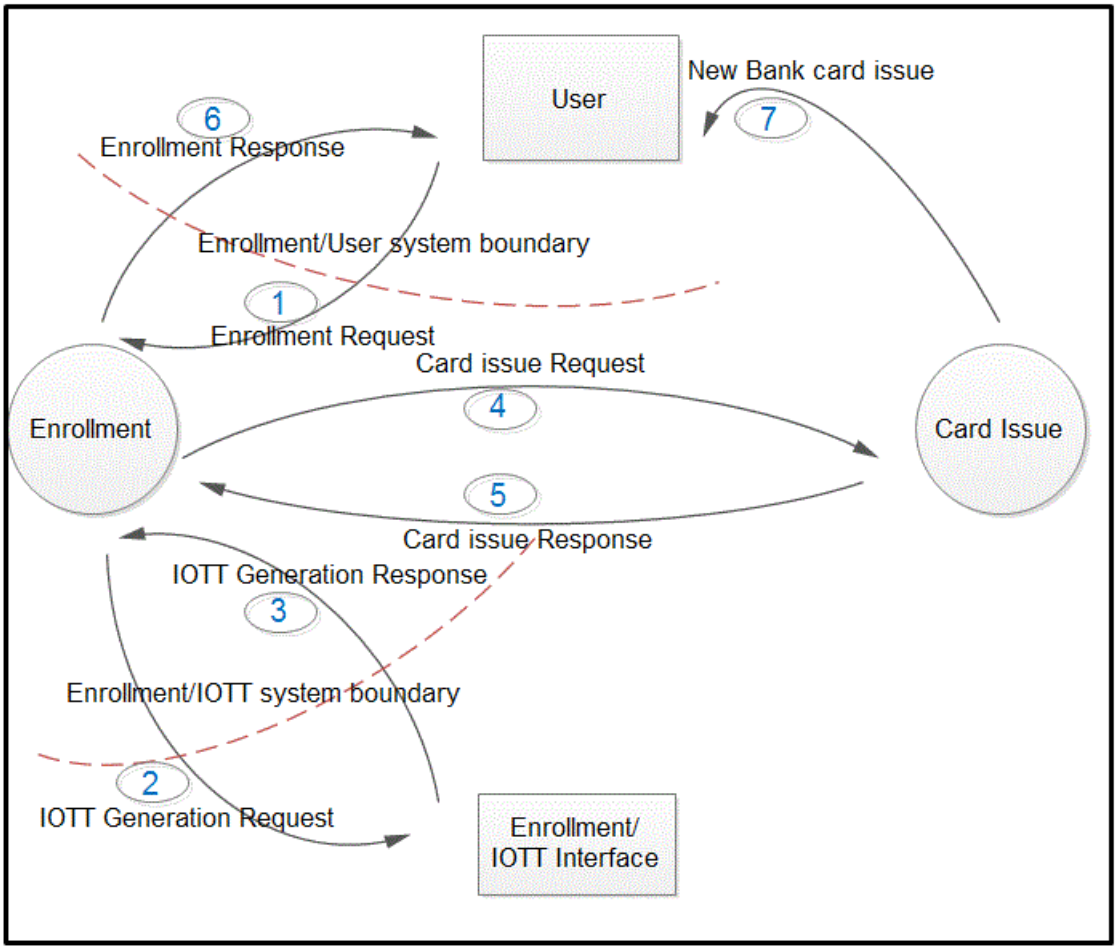


Figure 5.1: Enrollment and card issue phase

5.1.2 IOTT generation and storage

The IOTT generation and storage process is illustrated in Figure 5.2. The labels numbered in the figure represent the sequence of events in the IOTT generation and storage process. In this process, the *enrollment/IOTT interface* transmits the IOTT generation request (*IOTT gen. Request(1)*) to the *IOTT generator*. It first extracts the attribute list that is sent (*Send attribute list(2)*) to the *BK generator (Biometric Key generator)*, which then processes the attribute list and sends back the output as a set of BKs (*Send BKs(3)*) to the *IOTT generator*. The *BK generator* subsequently sends a request to the *Biometric Template Transformation (BTT)* module with the BKs (*Send BKs(4)*). This triggers the *BTT*, which in turn sends a request to the *IOTT generator* for the finger template feature vector (*Request for fv(5)*). The *IOTT generator* responds with the feature vector (*Respond with fv(6)*). The *BTT* accepts as input the fv and BKs and uses the transformation function to generate the *Transformed Templates (TTs)*. The TTs are subsequently sent to the *IOTT*

generator (*Send TTs (7)*). The *IOTT generator* generates the IOTTs for each transaction based on the specific BK and the Transaction Sequence Number (TSN). Upon the successfull generation of the IOTTs, the IOTTS, BKs, and TSNs are mapped under the user PAN received in the IOTT generation request. This data is stored in the *IOTT data store* (*PAN, IOTTs, BKs, TSNs(8)*). Finally, the *IOTT generator* sends the IOTT generation response (*IOTT gen. response(9)*) to the *enrollment/IOTT interface*.

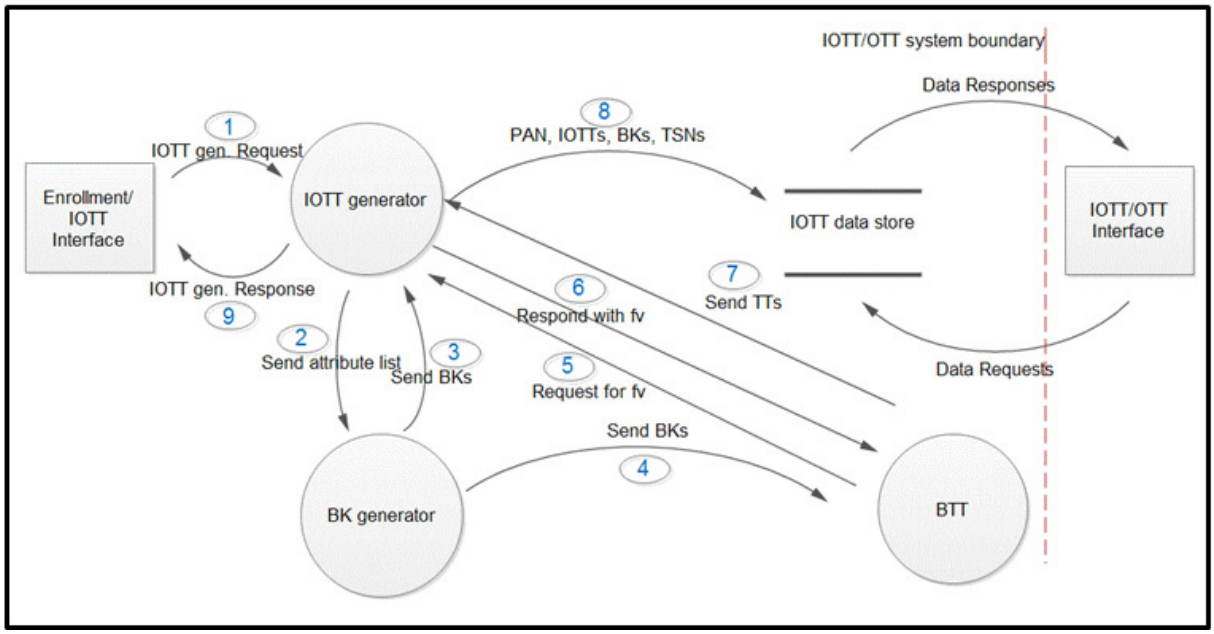


Figure 5.2: IOTT generation and storage phase

5.1.3 Transaction processing in POS

POS transaction processing is illustrated in Figure 5.3. The user is prompted to present the magnetic stripe bank card at the POS (*Request(1)*). The card is swiped (*Response(2)*), and the card details necessary to process the transaction, such as the user PAN and card expiry date are obtained by the *core POS transaction processor* module. This module then creates a logon request packet with the user PAN, date stamp, and time stamp, and then sends it (*Request(3)*) to the *message handler*. The *message handler*, on receiving the logon request packet assembles the message header and trailer. It then sends it to the *POS/comms. interface* (*Logon Request(4)*) and waits for the logon reply message (*Logon Reply(5)*) before proceeding with further transaction processing.

The *message handler* on receiving the logon reply message (*Logon Reply(5)*), triggers the *core POS transaction processor* through a response. At this stage, it is necessary to

authenticate the user to proceed with further transaction processing. It is the responsibility of the *security subsystem* to do the user authentication. The authentication process (*Request (7)* and *Response (8)*) between the POS and the BAS will be addressed in detail in section 5.1.4. Once the authentication process is successful, the rest of the transaction processing follows as normal in the current POS transaction system.

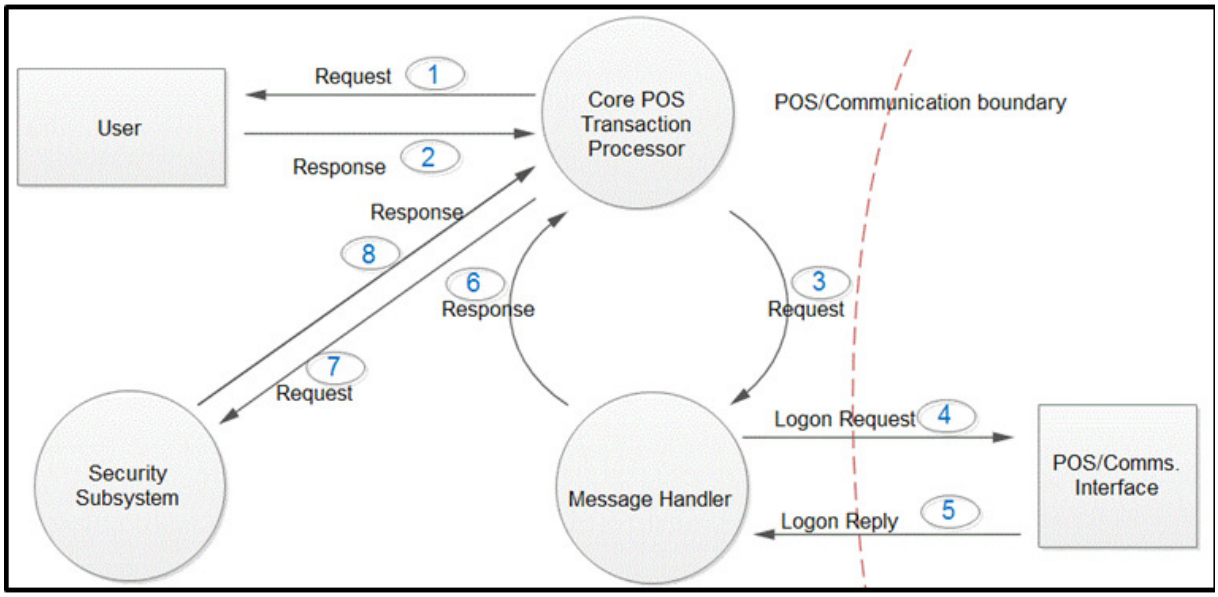


Figure 5.3: POS transaction processing phase

5.1.4 Authentication between POS and BAS

The authentication process between the POS and the BAS is illustrated as in Figure 5.4. The BAS is responsible for generating an OTT, which is used for authenticating the current transaction session. The OTT generation process was explained in detail in section 4.1.2 of Chapter 4. The generated OTT is stored in the *OTT data store*.

During the authentication phase, the *core POS transaction processor* requests the *user* to present the fingerprint (*Request(1)*) for authentication, at which point the *user* presents the fingerprint (*Reply(2)*). As described in the previous section, at the POS, the *core POS transaction processor* triggers the *security subsystem* to start with the authentication process (*Request(3)*). The *security subsystem* sends the reply (*Reply(4)*) to the *core POS transaction processor*.

The finger template is processed and the corresponding OTT is generated by the *core POS transaction processor*. The authentication request message is then assembled and is sent

(*Requests(5)*) to the *message handler*, which in turn sends it to the *POS/comms. interface (Authentication Request(6))*. This interface directs the message (*Request(7)*) to the *BAS* via the *comms. network*.

The *BAS* receives the authentication request via the *BAS/comms. interface* and passes the message (*Authentication Request(8)*) to its *message handler,* which routes the message (*Request(9)*) to the *core BAS transaction processor* for further processing. This process invokes (*Request(10)*) the *authentication subsystem*, which liaises with the *OTT data store* and retrieves the OTT necessary to authenticate the current transaction session through events *Data Storage Requests(11)* and *Responses(12)*. The retrieved OTT is then matched against the incoming OTT generated from the POS. The *authentication subsystem* now sends the response (*Reply(13)*) to the *core BAS transaction processor*. This process then assembles the authentication response message and sends (*Reply(14)*) it to the *message handler*, which in turn sends the message (*Authentication Reply(15)*) to the *BAS/comms. interface*. The *BAS/comms. interface* sends the reply (*Reply(16)*) to the *POS/comms. interface.*

The *message handler* module in the POS receives and processes the authentication response message (*Authentication Response(17)*), which then forwards the reply (*Reply(17)*) to the *core BAS transaction processor* for further processing. If the authentication phase succeeds, the normal POS transaction process follows, whereas if the authentication is unsuccessful, the transaction processing at the POS fails, and the user is informed accordingly.

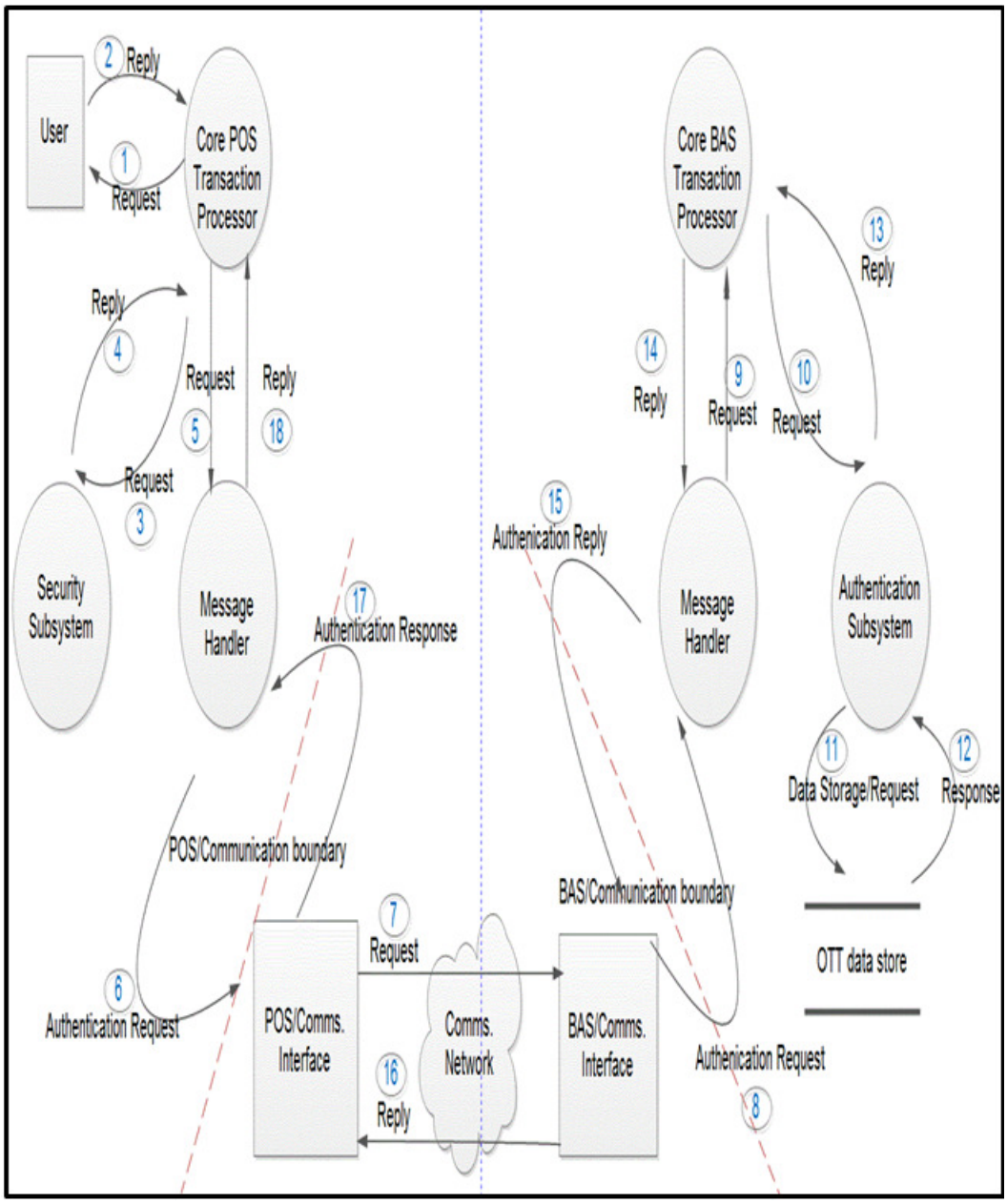


Figure 5.4: Authentication between POS and Server

Having modeled the core FAS processes and the data flow between them, the next section attempts to model the communication protocol of the PFAS.

5.2 PFAS communication protocol

The POS payment transaction chain differs from one retail environment to another. Normally, big retailers use a centralised POS back office server, which accepts all connections from different POS terminal tills and routes them to the retailer's acquiring bank, whereas small retailers connect the POS directly to the acquiring bank. The

acquiring bank is the bank or the financial institution or its agent that facilitates the EFT transactions for the retailer. The acquiring bank is referred to as the acquirer.

Acquirers typically make use of proprietary transaction protocols. These transaction protocols may differ in certain aspects. As an example, for certain acquireres, the authentication and the transaction processing may be combined, whereas it may be separated for other acquirers. The authentication process itself may be different, where an acquirer may either use separate servers for biometric authentication and PIN authentication, or use a single server to handle both types of authentication. The acquirer routes the transaction to the issuer's payment card network, which in turn connects to the respective issuing bank. An issuing bank holds each cardholder's bank account and is commonly known as the issuer. In some cases, the issuer and the acquirer can be the same bank and in this case routing is not required. Several variations of the payment chain are possible, depending on the corresponding retail environment and the underlying transaction protocol.

For simplicity and clarity purposes, the PFAS will be based only on the key entities in the POS transaction framework; which is the user, the POS, and the BAS. Hence, this section focuses on formulating a communication protocol for the PFAS with these key entities. The PFAS can be abstracted into any POS transaction and banking environment. The communication protocol of the PFAS is illustrated as in Figure 5.5.

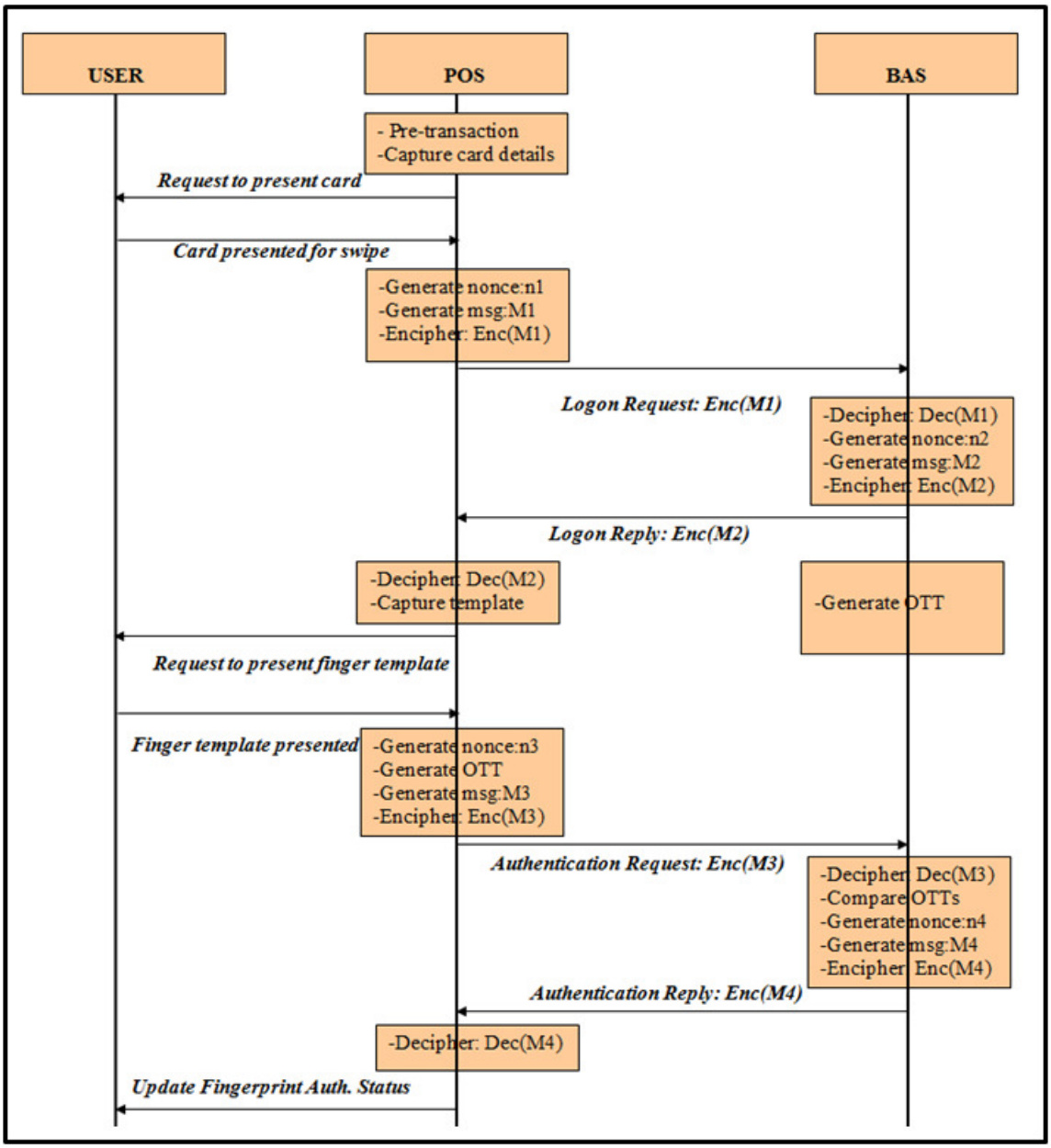


Figure 5.5: PFAS communication protocol

The security standards set in the protocol must comply with the Payment Card Industry Data Security Standards (PCI DSS). All the incoming and outgoing messages in the POS and BAS must be encrypted and authenticated [89]. The PCI DSS suggests the usage of ANSI X.94 Derived Unique Key Per Transaction (DUKPT) standard for authenticating and encrypting the messages [89, 90, 91]. In addition to this, PCI DSS mandates the usage of Transport Layer Security (TLS[4]) when transmitting sensitive data through public

[4] Transport Layer Security (TLS) (also known popularly as Secure Sockets Layer (SSL)) is a cryptographic protocol that is designed for Internet communication security. This protocol allows client-server applications to communicate across a network in a way designed to prevent eavesdropping and tampering.

communication channels [92]. Hence, the PFAS communication messages are based on the DUKPT standards using the TLS channel between the POS and the BAS. The POS is equipped with a Tamper-Resistant Security Module (TRSM). The retailer must be equipped with a secure environment, and the TRSM in the POS must be injected with the DUKPT data encryption keys which are necessary for authentication and encryption purposes. The BAS must also be equipped with a TRSM for encrypting and decrypting secret data. As explained in section 4.1.2 of Chapter 4, the BAS is equipped with an IOTT generation module, which generates the corresponding IOTTs for each user and for each TSN, based on the BKs. The IOTT generation process takes place during the enrollment and card issuing processes. The BK generation process is explained in detail in section 4.1.2 of chapter 4.

The protocol commences when the POS enters the pre-transaction phase. This phase starts with the POS prompting the user to present the card. The user presents his or her magnetic stripe bank card, which is swiped. The POS subsequently captures all the relevant card information. The POS then enters the pre-authentication phase and creates a message M1, which is a logon request message, encrypts it and sends it to the BAS. The message M1 consists of the nonce[5] n1, PAN, date stamp, and time stamp. The BAS on receiving M1, passes the encrypted message to the TRSM, which decrypts the message and passes it back to the BAS. The BAS checks the authenticity of M1. The authenticity check ensures that the contents of a message have not been tampered with and altered during the communication between the POS and the BAS. The authenticity is performed using the Message Authentication Code (MAC) technique (MAC is a short piece of information used to authenticate a message) [39]. After performing the authenticity check, BAS retrieves the TSN and the BK for the corresponding PAN. A nonce n2 is generated, and Message M2 is assembled with n2, TSN, and BK. The message is passed to TRSM for encryption and the encrypted message is sent back as a logon reply to the POS.

[5] In security protocols, a nonce is an arbitrary number or a challenge used only once in a cryptographic communication. It is basically a random or pseudo-random number generated with the purpose of being used only once in a single run of the protocol.

After sending the logon reply message, the BAS proceeds through the creation of the OTT as explained in detail in section 4.1.2 of Chapter 4. The OTT necessary for authenticating the current transaction session is generated. The POS on receiving the logon reply message decrypts it, and checks the authenticity of M2. It then generates an OTT by following the same process as in BAS. A nonce n3 is generated and message M3 is assembled with n3, OTT, and PAN. The authentication request message M3 is encrypted, and sent to the BAS. The BAS on receiving M3, passes the encrypted message to the TRSM, which decrypts the message and passes back to the BAS. The BAS first checks the authenticity of M3. It then matches the incoming OTT with the generated OTT at the BAS for the incoming PAN. If the score is within an acceptable threshold, the authentication is successful. Otherwise, the authentication fails. The acceptable threshold referred here is the value that allows the system to differentiate between a genuine user and an imposter [93]. The acceptable threshold value that is calibrated for PFAS is at 0.48 (the details are provided under subsection 7.2.1.1 of Chapter 7).

The BAS subsequently generates n4 and assembles the authentication reply message M4 with n4 and the authentication result. The message M4 is transmitted to TRSM for encryption and the BAS ultimately sends the encrypted message to the POS. The POS, on receiving M4 decrypts it. It first checks the authenticity of M4 and makes the decision on the authenticity of the card based on the authentication result field. If the authentication result is successful, the POS determines that the user's bank card is genuine and is not a cloned card. The POS then continues with the existing transaction processing, which is identical to the processing that is currently done in the retail environment and falls outside the scope of the current study. If the authentication result is a failure, the POS determines that the user is using a cloned magnetic stripe bank card. Hence, the transaction processing will be stopped at the POS and the user will be informed accordingly.

5.3 Summary

This chapter provided a thorough analysis and design of the PFAS in order to address the research problem. The individual building blocks of the PFAS and their interaction were designed using the SSADM methodology. DFD models were created for all processes in the PFAS. The last section of the chapter presented the core communication protocol that drives the PFAS.

6
Verification and validation

Chapter 5 designed the individual entities in the PFAS and derived the PFAS communication protocol. The aim of this chapter is to verify and validate the PFAS. For this purpose, the current chapter is divided into three main subsections. Section 6.1 studies the existing security protocol modelling tools prior to selecting an appropriate modelling tool for verifying the PFAS. Section 6.2 considers the verification of the PFAS, whereas section 6.3 considers its validation. The chapter is concluded in section 6.4.

6.1 Selecting a modelling tool for the PFAS

A plethora of methodologies exists for verifying security protocols. These methodologies range from manual to formal verification techniques. The major techniques are classified as equivalence checking, model checking, and theorem proving techniques. The research within the field of security protocol verification is a productive area because security protocols are often prone to errors, and it is not easy to identify errors through manual verification procedures. Due to this shortcoming, automatic verification tools are extensively used for verification purposes. Well-known automatic tools, such as ProVerif, Capser, and Avispa are commonly used for evaluating security protocols. Table 6.1 illustrates various security protocols and their corresponding verification tools.

Table 6.1: Security protocols and verification tools

Security protocols	**Verification tools used**	**References**
TLS, Otway-Rees, Needham-Schroeder	HOL/ Isabelle	[94]
Wide-mouthed-frog-protocol	FDR/Casper	[95]
TLS	Avispa	[96]
E-Voting (FOO92), JFK	ProVerif	[97, 98]

In a study conducted by Cas Cremers and Pascal Lafourcade in their research paper "Comparing State Spaces in Automatic Security Protocol Verification", six verification tools were compared against each other. Avispa consists of four tools, which are TA4SP, CL-Atse, OFMC, and Sat-Mc. Avispa tools, ProVerif, and Scyther were compared in their study and the security properties of each tool were modeled [99]. In each tool, the secrecy of nonce and session key were analysed, and the performance of each tool was evaluated. This study came to the conclusion that ProVerif is the fastest tool in terms of the time required to verify the security properties, and the most successful in analysing cryptographic protocols [99]. The applicability of ProVerif has been widely validated. The following table includes a list of key security protocols in the literature that have been successfully verified using ProVerif [100].

Table 6.2: List of key security protocols verified using ProVerif

Security Protocols	**References**
Needham-Schroeder	[101]
Woo-Lam	[102]
Denning-Sacco	[103]
Yahalom	[104]
Otway-Rees	[105]

ProVerif has also been extensively used in numerous case studies of security protocols. The most significant of these case studies are listed in what follows.

- Blanchet and Chaudhuri studied the integrity of the Plutus file system on untrusted storage, which led to the discovery and the resolution of the security vulnerabilities in the original system [106, 107].
- Abadi and Blanchet deployed correspondence assertions to evaluate the certified email protocol [108].
- Bhargavan used ProVerif to analyse cryptographic protocol implementations written in F#, and the TLS protocol has been analysed in this manner [109, 110, 111, 112].
- Abadi, Blanchet, and Fournet evaluated the Just Fast Keying (JFK) protocol using ProVerif [113, 114].

- Delaune, Kremer, and Ryan, as well as Backes, Hritcu, and Maffei formalised and evaluated privacy properties for electronic voting using ProVerif [115, 116, 117].
- Delaune, Ryan, and Smyth, as well as Backes, Maffei, and Unruh analysed the anonymity properties of the trusted computing scheme Direct Anonymous Attestation (DAA) using ProVerif [118, 119, 120].
- Kusters and Truderung examined with Diffie-Hellman exponentiation and XOR (Exclusive OR) using ProVerif [121].
- Chen and Ryan evaluated authentication protocols found in the Trusted Platform Module (TPM) and uncovered several security issues through ProVerif [122].
- Smyth, Ryan, Kremer, and Kourjieh evaluated the electronic voting protocol using ProVerif [123, 124].

The next subsection provides a detailed overview of the ProVerif tool.

6.1.1 Understanding ProVerif

ProVerif is well-known for modeling and analysing security protocols [125, 126]. It was developed by Bruno Blanchet and is dedicated to proving secrecy, authenticity, and other properties of security protocols [97]. It accepts as input a set of queries and outputs true or false for each query. The queries are first translated to a set of clauses [127, 128]. This will yield an abstract representation of the protocol, and ultimately ProVerif aims to resolve these clauses.

Processes are the key entities in ProVerif, and they are represented using a set of *names*, *variables,* and *function symbols* [127]. The *names* are used to represent constants or communication channels, and the function symbols are used to represent terms. Terms are used to define names, variables, and function symbols, which can also be applied to other terms [127]. Typical function symbols such as **enc** for *encryption* and **dec** for *decryption* are included in ProVerif and it also facilitates the definition of equations, which are typically based on terms constructed from the function. A sample equation showing the encryption/decryption function is as follows:

equation dec (enc (x, k), k) = x **(2)**

In equation (2), the **enc** takes plain text x and a key k, and returns the corresponding cipher text. The **dec** function takes cipher text and a key k and returns the plain text x. ProVerif has a family of proof techniques and has been used to analyse and prove a variety of security protocols [94, 95, 96, 129]. Hence, ProVerif is selected to model the PFAS.

Danny Dolev and Andrew C. Yao in their research work, "On the Security of Public Key Protocols", introduced a set of attack models to evaluate the security of a family of communication and security protocols [130]. ProVerif is compliant with the Dolev-Yao model and hence can model the attack scenarios. In ProVerif, the query commands are formulated to evaluate the properties of a protocol. As an example, the query '*attacker: m'* will be satisfied if an *attacker* may obtain the message *m* by issuing the messages on public channels and by applying functions to them. ProVerif in some cases can also prove that the processes are observationally equivalent [131]. The keywords of the ProVerif tool are as follows [131]:

among, and, choice, clauses, data, elimtrue, else, equation, event, free, fun, if, in, let, new, noninterf, not, nounif, out, param, phase, putbegin, pred, private, process, query, reduc, suchthat, then, and *weaksecret*.

The ProVerif grammar is illustrated in Figure 6.1.

P, Q, R processes

0 null process

P |Q parallel composition

new n; P name restriction

new x; P variable restriction

equation <term >= <term > the terms M1 and M2 are in fact equal

query attacker: M determines whether the attacker may have M.

if M = N then P else Q conditional

event x; P event launch

let x = M in P replace the variable x with the term M in process P

in(M,N); P message input

out(M,N); P message output

!P replica

Figure 6.1: ProVerif grammar [131]

The ProVerif grammar is summarized as follows:

- ***equation <term >= <term >*:** equation M1 = M2 implies that terms M1 and M2 are in fact equal. The function symbols in the equation should be the constructors that were declared previously.
- ***query attacker*:** M determines whether the attacker may have M.
- ***not attacker* :** M is true when M is a secret.
- ***if f then P else Q*:** This test executes P when the fact is true. Otherwise, it executes Q. The process if f then P is equivalent to *if f then P else 0*.
- ***event M;P*:** The event command emits the event M and then executes P.
- ***let p = M in P*:** The let command executes P after matching the term M with the pattern p. If the term M does not match the pattern p, the process blocks.
- ***in(c,p); P*:** The input command inputs a message on channel c, and executes P after matching the input message with p.
- ***out(c,p); P*:** The output command outputs a message on channel c, and executes P after sending the message.

This section provided a detailed overview of the ProVerif tool, its syntax, and grammar. Further details may be obtained by referencing the ProVerif User Manual (reference [131] in the Bibliography section).

Having concluded that ProVerif is the most suitable for modelling the PFAS, the next section concentrates on the verification aspects using ProVerif.

6.2 Verification

Verification is defined as "the process of determining if the particular implementation of a model or a simulation accurately represents the conceptual description and specifications" [132]. It questions whether a model or a simulation works as intended. Stated differently, it determines whether the requirements for a system are accurate and complete.

6.2.1 Verifying the PFAS using the ProVerif model

To ensure that the PFAS provides the appropriate level of security and the objectives that it upholds in addressing the research problem, the ProVerif model will be used in the verification. The main block diagram of the PFAS is presented in Figure 6.2.

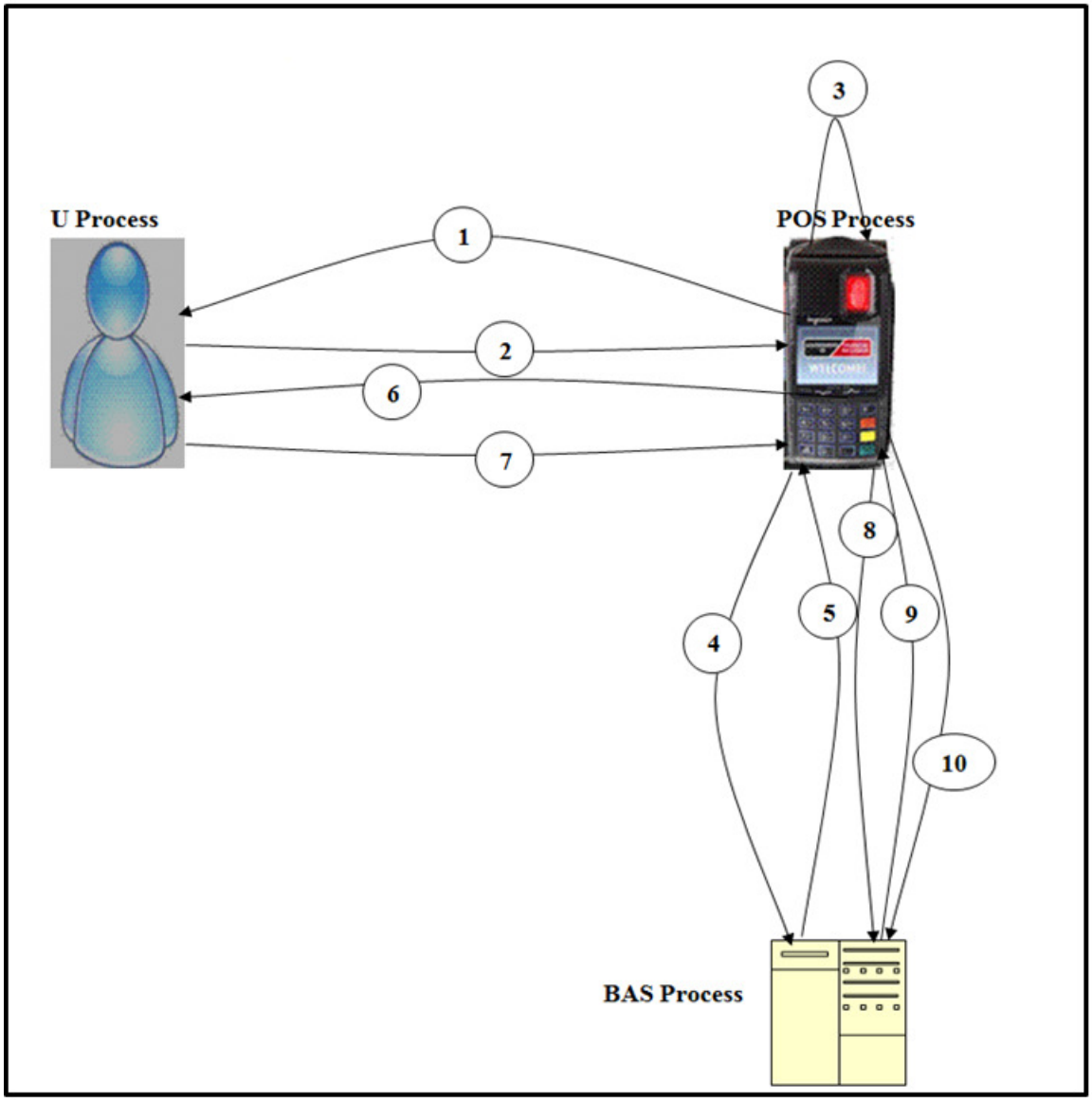


Figure 6.2: ProVerif model of the PFAS

The labels illustrated in the Figure 6.2 correspond to ProVerif events or messages between the processes in the PFAS. These messages are as provided in Table 6.3.

Table 6.3: Label to Event/Message Mappings

Label	Event
1	PresentCard
2	CardPresented
3	CardSwiped
4	LogonRequest
5	LogonResponse
6	PresentFingerPrint
7	FingerPrintPresented
8	AuthenticationRequest
9	AuthenticationResponse
10	AuthenticationStatusUpdate

The current model incorporates a *U Process*, *POS Process,* and *BAS Process*. These processes are subsequently referred to as *UP*, *PP,* and *BP*, respectively. The communication between the processes is based on the key events or messages listed in Table 6.3. The details regarding each individual building block of the model are provided in the following subsections.

6.2.1.1 U Process (UP)

The *UP* is responsible for all activities pertaining to the user. A script is developed in the ProVerif grammar to verify the *UP*. It is presented in Table 6.4 in the appendix. The comments in the script are surrounded by (* and *) and nested comments are not supported. The script uses *c* to denote the public channel and *bc* to denote the private or bounded channel, as declared in lines 7 and 10 respectively. Lines 13, 16, 19, 22, and 25 declare the following variables.

- *Magnetic Stripe Bank Card Data (MSCD)*
- *Fingerprint Template Request (FPReq)*
- *Fingerprint Template Presented (FPPresented)*
- *Authentication Status Update Success (ASUSuccess)*
- *Authentication Status Update Failure (ASUFailure)*

The event declarations are carried out from lines 28 to 32 and they represent the following events.

- e*ventPresentMSC(PresentMagneticStripeBankCardevent)*
- *eventCardSwipeFailure(CardswipeFailureevent)*
- *eventPresentFP(PresentFingerprintevent)*
- *eventAuthenticationResponseSuccess(AuthenticationResponseSuccessevent)*
- e*ventAuthenticationFailure(AuthenticationResponseFailureevent)*

The main process starts on line 51, and on line 54 the *MSCD* is sent out via the *bc* channel. Line 57 represents the reception of a message via the incoming channel. On line 60, it is determined whether *x* corresponds to *FPReq*; if this is the case, it is asserted that the events *evPresentMSC* and *evPresentFP* must have occurred. In line 64, the message *FPPresented* is sent out via the outgoing channel. Line 64 indicates that if *x* does not correspond to an *FPReq*, then only the *evPresentMSC* and *evCardSwipeFailure* events could have occurred prior to the reception of the message.

Line 70 represents the reception of a message via the incoming channel. On line 73 it is determined whether *y* corresponds to an *ASUSuccess*; if this is the case, then it is asserted that events *evPresentMSC, evPresentFP,* and *evAuthenticationResponseSuccess* must have occurred. On line 64, the message *FPPresented* is sent out via the outgoing channel. Line 77 indicates that if *y* does not correspond to an *ASUSuccess*, then only the *evPresentMSC, evPresentFailure,* and *evAuthenticationFailure* events could have occurred prior to the reception of the message. The next section discusses the *PP*.

6.2.1.2 POS Process (PP)

The *PP* interfaces with the *UP* and *BP* in facilitating the transaction process. A script was developed to verify the *PP*. It is presented in Table 6.5 in the appendix. Lines 13, 16, 19, 22, 25, 28, and 31 declare the following variables.

- *MSCD*
- *LogonRequest(LReq)*
- *LogonResponseSuccess(LRespSuccess)*
- *FingerPrintRequest(FPReq)*
- *FingerPrintPresented(FPPresented)*
- *AuthenticationRequest(AReq)*

- *AuthenticationResponseSuccess(ARespSuccess)*

The event declarations are carried out from line 34 to 43 and they represent the following events:

- *evMSCPresented*
- *evCardSwiped*
- *evCardSwipeFailure*
- *evLogonResponseSuccess*
- *evLogonResponseFailure*
- *evFPPresented*
- *evAuthenticationRequestSuccess*
- *evAuthenticationFailure*
- *evTransactionDeclined*
- *evTransactionContinue*

The main process starts on line 76. Line 79 represents the reception of a message via the incoming channel. On line 81, it is determined whether *w* corresponds to *MSCD* and, if this is the case, a *Logon Request (LReq)* is sent through the bounded channel to *BP*. All the message exchanges between *PP* and *BP* are secure and they are generated, encrypted, and decrypted as explained in section 5.2 of Chapter 5. Line 84 indicates that if *x* does not correspond to an *MSCD*, then *UP* is informed about the *CardSwipeFailure* on line 86.

On line 89, *PP* receives an incoming message from *BP*, whereas on line 91, it is determined if *x* corresponds to a *successful Logon Response (LRespSuccess)*. If this is the case, then it is determined on lines 92 to 94 whether the events *evLogonResponseSuccess*, *evCardSwiped,* and *evMSCPresented* occured. After the event checks, the *request to present fingerprint(FPReq)* is sent through the bounded channel to *UP* on line 96. Line 97 indicates that if *x* does not correspond to *LRespSuccess*, then the event checks are done on lines 98 through 101 for *evTransactionDeclined*, *evLogonResponseFailure*, *evCardSwiped,* and *evMSCPresented*.

Line 104 represents the reception of a message via the incoming channel and on line 106, it is determined whether *y* corresponds to *FPPresented*. If this is the case, then an *AuthenticationRequest(AReq)* message is sent to *BP* on line 108. Line 114 represents the reception of the message via the incoming channel and on line 117, it is determined whether *z* corresponds to a *successful Authentication(ARespSuccess)*. If this is the case,

then it is determined on lines 118 to 123 whether the events *evTransactionContinue, evAuthenticationResponseSuccess, evLogonResponseSuccess, evFPPresented, evCardSwiped, and evMSCPresented* occured. Line 125 indicates that if *z* does not correspond to an *ARespSuccess*, then the event checks are done on lines 125 to 130 for *evTransactionDeclined*, *evAuthenticationFailure*, *evLogonResponseSuccess*, *evFPPresented*, *evCardSwiped,* and *evMSCPresented*. The next section discusses *BP*.

6.2.1.3 BAS Process (BP)

The *BP* is primarily responsible for the course of actions involved in authenticating the requests from the *PP*. A script was developed to verify the *BP*. It is presented in Table 6.6 in the appendix. Lines 13, 16, 19, 22, 25, and 28 declare the following variables.

- *ValidLogonRequest(VLReq)*
- *LogonResponseFailure(LRespFailure)*
- *FingerPrintRequest(FPReq)*
- *FingerPrintPresented(FPPresented)*
- *ValidAuthenticationRequest(VAReq)*
- *AuthenticationResponseSuccess(ARespSuccess)*
- *AuthenticationResponseFailure(ARespFailure)*

The event declarations are carried out from line 31 to 36 and they represent the following events.

- *evLogonRequest*
- *evLogonResponseSuccess*
- *evLogonResponseFailure*
- *evAuthenticationRequest*
- *evAuthenticationResponseSuccess*
- *evAuthenticationFailure*

The main process starts on line 56. Line 59 represents the reception of a message via the incoming channel and on line 61, it is determined whether *x* corresponds to *VLReq*. If this is the case, the event *evLogonRequest* is run on line 62, after which a *LRespSuccess* message is sent through the bounded channel on line 64. Line 65 indicates that if *x* does not correspond to a *VLReq*, then the event *evLogonResponseFailure* is run on line 66 and the *PP* is informed about the *LRespFailure* through the outgoing message on line 68.

Line 72 represents the reception of a message via the incoming channel and on line 74, it is determined whether *y* corresponds to *VAReq*. If this is the case, the event *evAuthenticationResponseSuccess* is run on line 75, after which an *ARespSuccess* is sent through the bounded channel to *PP* on line 77. Line 78 indicates that, if *y* does not correspond to a *VAReq*, then the event *evAuthenticationFailure* is run on line 79 and *PP* is informed about the *ARespFailure* through the outgoing message on line 81.

The verification of the PFAS was conducted in two phases. In the first phase, an extensive code review was conducted against each script in the model (presented in the appendix section in Tables 6.4, 6.5, and 6.6). In the second phase, the scripts were sequentially executed against the ProVerif modeling tool. The verification results are presented as screenshots in Figures 6.3, 6.4, and 6.5.

```
C:\ProVerif>proverif ./FASP/UProcess.pv
Process:
(1)out(bc, MSCD);
(2)in(bc, x: bitstring);
(3)if (x = FPReq) then
    (4)event evPresentFP;
    (5)event evPresentMSC;
    (6)out(bc, FPPresented)
else
    (7)event evCardSwipeFailure;
    (8)event evPresentMSC;
    (9)in(bc, y: bitstring);
    (10)if (y = ASUSuccess) then
        (11)event evAuthenticationResponseSuccess;
        (12)event evPresentFP;
        (13)event evPresentMSC
    else
        (14)event evAuthenticationFailure;
        (15)event evPresentFP;
        (16)event evPresentMSC

-- Query event(evAuthenticationFailure) ==> (event(evPresentFP) && event(evPrese
ntMSC))
Completing...
Starting query event(evAuthenticationFailure) ==> (event(evPresentFP) && event(e
vPresentMSC))
goal reachable: begin(evPresentMSC) -> end(evAuthenticationFailure)

1. The message MSCD[] may be sent on channel bc[] at output (1).
mess(bc[],MSCD[]).

2. The message MSCD[] that may be sent on channel bc[] by 1 may be received at i
nput (2).
The event evPresentMSC may be executed at (8).
The message MSCD[] that may be sent on channel bc[] by 1 may be received at inpu
t (9).
We have MSCD[] <> ASUSuccess[] & MSCD[] <> FPReq[].
So event evAuthenticationFailure may be executed at (14).
end(evAuthenticationFailure).


Could not find a trace corresponding to this derivation.
RESULT event(evAuthenticationFailure) ==> (event(evPresentFP) && event(evPresent
MSC)) cannot be proved.
-- Query event(evAuthenticationResponseSuccess) ==> (event(evPresentFP) && event
(evPresentMSC))
Completing...
Starting query event(evAuthenticationResponseSuccess) ==> (event(evPresentFP) &&
 event(evPresentMSC))
RESULT event(evAuthenticationResponseSuccess) ==> (event(evPresentFP) && event(e
vPresentMSC)) is true.
-- Query not attacker(FPPresented[])
Completing...
Starting query not attacker(FPPresented[])
RESULT not attacker(FPPresented[]) is true.
-- Query not attacker(MSCD[])
Completing...
Starting query not attacker(MSCD[])
RESULT not attacker(MSCD[]) is true.
```

Figure 6.3: Verification of the U Process

Figure 6.4: Verification of the POS Process

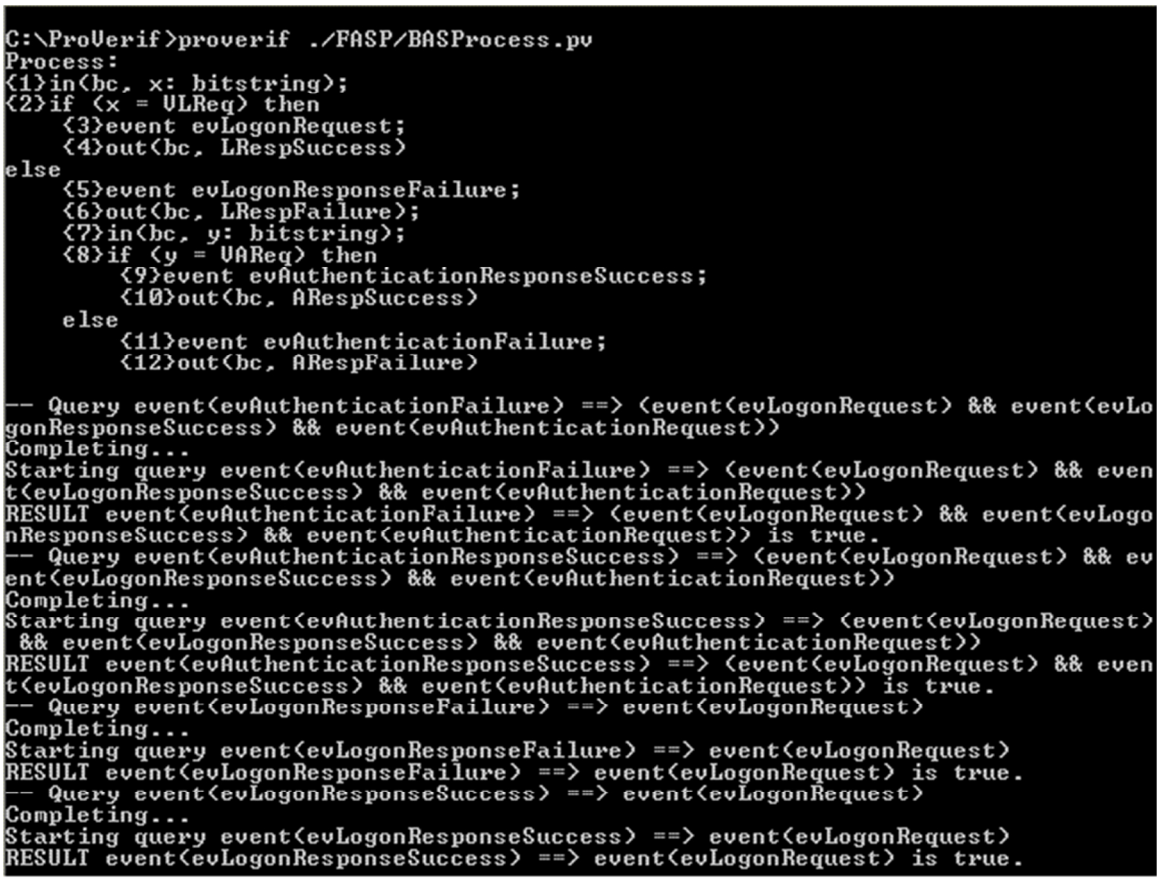


Figure 6.5: Verification of the BAS Process

The verification conducted in the first and second phase revealed that all the requirements imposed on the PFAS for successfully addressing the research problem are implemented correctly. Further, they are represented precisely as according to the PFAS requirements and design conducted in Chapter 5. Having discussed the verification part, the next section focuses on the validation.

6.3 Validation

Validation is defined as "the process of determining the degree to which a model or simulation is an accurate representation of the real world from the perspective of the intended use of a model or a simulation" [132]. Validation questions whether a model or a simulation is indeed realistic.

This section validates whether the PFAS achieves its key objectives (as listed in Figure 6.6) and ultimately solves the research problem. Each of these objectives is analysed in the following subsections. It should be noted that, although each of these objectives are

addressed separately, they are interdependent in many ways; as a result, the discussions of the objectives tend to overlap to some degree. In order to validate whether the model meets its intended objectives, various attacks on the PFAS are generated in the scripts corresponding to each process. The objectives are analysed in terms of the *event* and *query commands*. The *event command* is used to launch an event when a specific action is executed, whereas the *query command* is used to prompt ProVerif to validate the correctness of the sequence of specified events. If it is determined that the sequence is not correct, ProVerif declares that an attack has been identified. As with any security measurement tool, ProVerif can validate only generic security objectives. Therefore, the objectives that are specific to the PFAS will be analysed based on logical propositions and facts.

- Preserve the privacy and security of sensitive data.
- Mutual authentication.
- Resilience to the compromise of finger template.
- Support revocation of the template; in case it gets compromised.
- Resilience to replay attacks.

Figure 6.6: Key objectives

6.3.1 Privacy and security

The privacy and security of the finger template data are of utmost importance to the PFAS, as the compromise of the template data will be of permanent nature [70]. Security implies that data is readily available to authorised people and is unavailable to unauthorised people. Privacy reduces the pool of authorised people to those individuals who have a valid need to access the data. There is a truism that "*You can have security without privacy, but you cannot have privacy without security*" [133]. This implies that the confidentiality of template data must be protected to ensure privacy, and that security mechanisms are required in order to provide this protection. In the PFAS model, the following queries are executed in *UP* (Table 6.4, line number 35 to 38) and *PP* (Table 6.5, line numbers 46 to 49).

- *query attacker (MSCD).*
- *query attacker (FPPresented).*

The test results of these queries are illustrated in Figure 6.7, and discussed in what follows.

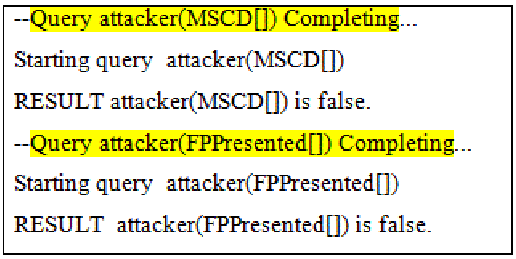


Figure 6.7: Test results 1

The results of the queries *attacker (MSCD)* and *attacker (FPPresented)* are *false*, which imply that they are not attacks. This indicates that the magnetic stripe card presented during the transaction is genuine. If the results of the queries were *true*, then it would have implied that the queries were attacks, and that the magnetic stripe card presented during the transaction was a cloned card. Hence, the results reveal that the privacy and security property are kept intact in the PFAS, and that it does not lead to a compromise. The next section evaluates the second security property in the list, which is the mutual authentication.

6.3.2 Mutual authentication

Authentication occurs after identification and before authorisation. It validates the authenticity of the identity declared during the identification phase. Mutual authentication implies the act of two parties thoroughly authenticating each other [133]. In the PFAS, the processes authenticate each other by using events to ascertain the mutual authentication. Hence, the relevant queries were written to test the processes in order to establish whether one event is not executed before another event or a group of events, and whether the mutual authentication was carried out as expected. The following queries were executed in the *UP* (Table 6.4, line numbers 42 to 47).

- *query event (evAuthenticationResponseSuccess) ==> event (evPresentFP) && event (evPresentMSC).*
- *query event (evAuthenticationFailure) ==> event (evPresentFP) && event (evPresentMSC).*

The test results of these queries are presented in Figure 6.8.

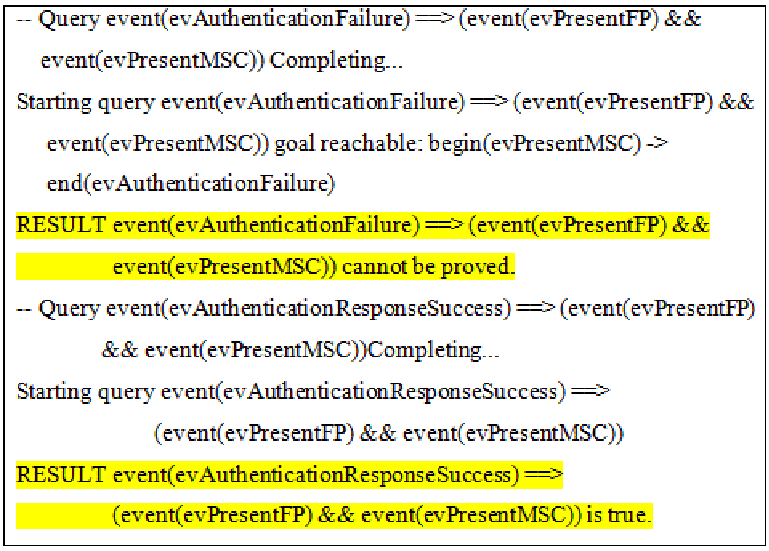


Figure 6.8: Test results 2

The results of the queries reveal that mutual authentication is successfully carried out in the *UP*. The following queries were executed in the *PP* (Table 6.5, line numbers 53 to 73).

- *query event (evLogonResponseSuccess) ==> event (evCardSwiped) && event (evMSCPresented).*
- *query event (evTransactionDeclined) ==> event (evLogonResponseFailure) && event (evCardSwiped) && event (evMSCPresented).*
- *query event (evTransactionContinue) ==> event (evAuthenticationResponseSuccess) && event (evLogonResponseSuccess) && event (evFPPresented) && event (evCardSwiped) && event (evMSCPresented).*
- *query event (evTransactionDeclined) ==> event (evAuthenticationFailure) && event (evLogonResponseSuccess) &&event (evFPPresented) && event (evCardSwiped) && event (evMSCPresented).*

The test results of these queries are presented in Figure 6.9.

Figure 6.9: Test results 3

The results of the queries reveal that the mutual authentication is successfully carried out in the *PP*. The following queries were executed in the *BP* (Table 6.6, line numbers 40 to 54).

- *query event (evLogonResponseSuccess) ==> event (evLogonRequest).*
- *query event (evLogonResponseFailure) ==> event (evLogonRequest).*
- *query event (evAuthenticationResponseSuccess) ==>*
  event (evLogonRequest) && event (evLogonResponseSuccess)
  && event (evAuthenticationRequest).
- *query event (evAuthenticationFailure) ==> event (evLogonRequest)*
  && event (evLogonResponseSuccess) && event (evAuthenticationRequest).

The test results of these queries are presented in Figure 6.10. The results of the queries reveal that mutual authentication is successfully carried out in the *BP*. Since all the processes successfully carry out this security property, it is established that this security property remains intact in the PFAS.

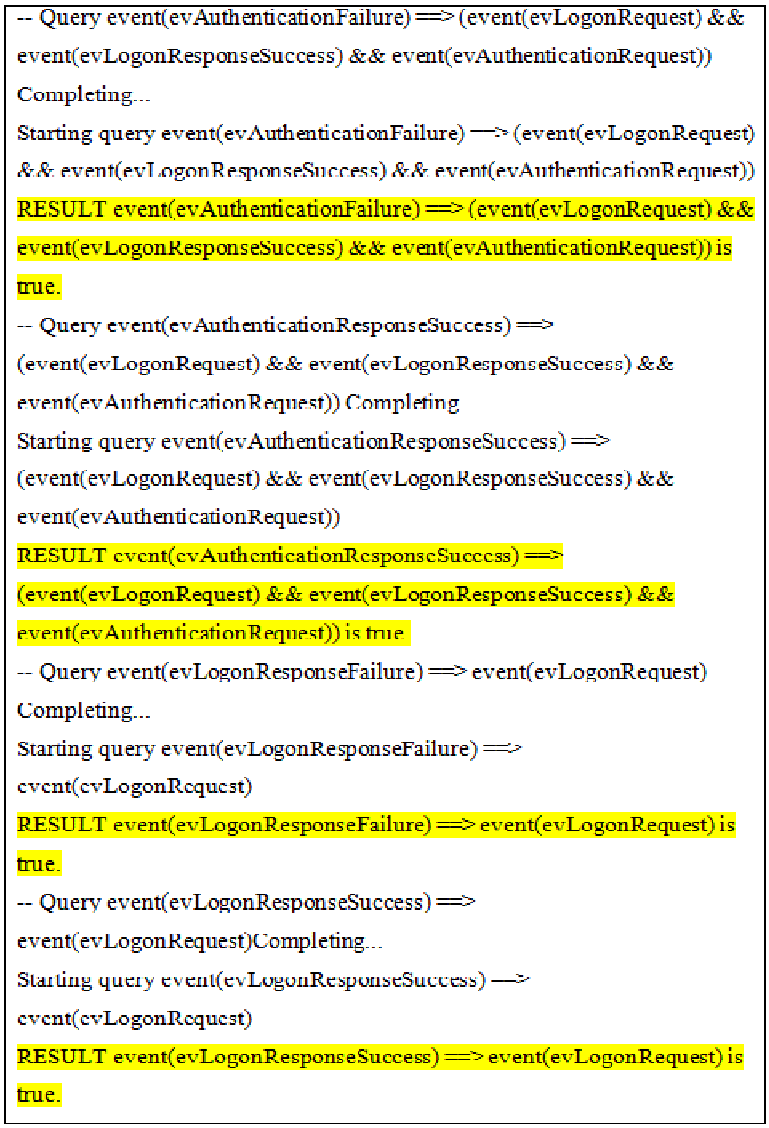


Figure 6.10: Test results 4

The next section validates the third security property, which is to achieve resilience to the compromise of the finger template.

6.3.3 Resilience to the compromise of finger template

The term "compromise" is loosely used in cryptography to imply that a password or a token has been exposed [133]. In biometrics, it has a different meaning and consists of three components. First, the attacker has to possess a reproduction of the biometric

template. Secondly, in order to make it practical, the attacker must have the knowledge and technology to be able to use it in a biometric authentication system. Thirdly, the attacker must be capable of mitigating any countermeasures that are applied to prevent its use [133].

In the PFAS, a unique OTT is generated in the POS and the BAS during each authentication session. As the name implies, an OTT can be used only once. Furthermore, the original finger template is not stored anywhere, and hence the current scheme is resilient to the compromise of the original finger template data. The following section evaluates the next security property on the list, which is the support for revocation.

6.3.4 Revocation support

According to ITU-T X.811[6], the definition of revocation is the "permanent invalidation of verification and authentication information" [134]. In the PFAS, even if an OTT is compromised, it is still revocable and replaceable. The original template of the user never leaves the POS or gets stored in the BAS. This offers more significant benefits to the users in terms of privacy and security. After the necessary transformation, the finger template is intentionally distorted to an IOTT and then to an OTT. These new versions of the finger template are secure, as the original fingerprint pattern cannot be reverse engineered from the OTT used during the authentication phase. They are also cancelable, as a totally different pattern or code can be generated using the transformation process. This is done by using a different finger template of the user or a different combination of attributes captured during the re-enrollment. Using this technique, one or two fingerprints can be mapped to a total of *n* different virtual IOTTs, thereby fulfilling the objective of the revocation support in the case of a compromised template. The next section validates the fifth security property on the list, which is the resilience to replay attacks.

6.3.5 Resilience to replay attacks

A replay attack is a two or three-stage process; first intercepting or copying the sensor transmission, then possibly modifying the data, and finally replaying the signal [135]. In

[6] ITU-T X.811 is the specification for an open systems interconnection authentication framework.

certain replay attacks, a false data stream may be injected between the sensor and the processing system. In most cases, this involves some physical tampering with the system [135]. If the true fingerprint is disclosed in the conventional FAS protocol, then the FAS is vulnerable to a replay attack. In the worst case, the same fingerprint could be used to illegally gain access to multiple databases, and database cross matching could be carried out to gather business intelligence [94].

As explained in section 4.1.2 of Chapter 4, the enrollment entity stores neither the original finger template nor the users' attribute list during the fingerprint enrollment process. By not storing this information, the possibility of future replay attacks is eliminated. In addition, the PFAS generates an OTT for each authentication session based on the OTT algorithm, as illustrated in Figure 4.7 of Chapter 4. Hence, even if one OTT template in one authentication session is compromised, it cannot be reused to launch a replay attack for future authentication sessions as the OTT is unique for each authentication attempt. As a result, the replay of the template does not result in successful authentication in the PFAS.

Having validated the PFAS in the above section, the following section thus summarises the current chapter.

6.4 Summary

After considering various security protocol verification models, this chapter identified the ProVerif model as a suitable model for verifying the PFAS. Furthermore, the verification and validation of PFAS were conducted. In the verification phase, a formal ProVerif verification model was created for the PFAS. The scripts were coded for each process, and they were verified through ProVerif. The results generated by ProVerif for each process were analysed and successfully validated to prove that the key objectives were indeed met in resolving the research problem. The next chapter focuses on the evaluation of the PFAS.

7
Evaluation of the PFAS

Chapter 6 conducted a comprehensive verification and validation of the PFAS. The current chapter concerns the evaluation of the PFAS, and is divided into three main subsections. Section 7.1 aims at the development of a PFAS simulator. Section 7.2 formulates the PFAS test strategy and identifies the relevant test cases. Further, the PFAS is thoroughly tested, the results captured and significant graphs are plotted from the test results. Section 7.3 focuses on the evaluation of the PFAS to determine if it is indeed capable of resolving the research problem. Finally, the chapter is concluded in section 7.4.

7.1 Development of the PFAS simulator

In order to evaluate the PFAS, it is essential to develop a PFAS simulator. The following subsection explains the implementation plan.

7.1.1 Implementation plan

Matlab is a dynamic numeric scripting language that is used globally by students, engineers, and scientists. Matlab is ideal for simulation and prototyping of a framework, an algorithm or an idea because of its flexible syntax, rich set of built-in functions, and language capabilities [136]. Hence, Matlab was selected as the appropriate tool for implementing the PFAS simulator. The PFAS simulator is implemented under the system configuration illustrated in Figure 7.1 and it is recommended to follow more or less the same system configuration for simulation purposes.

Matlab: Version R2012b with Image Processing Toolkit

Operating System : Windows 7 Enterprise Edition

Processor : Intel(R) Core(TM) i7-3470QM CPU @2.70GHz 2.70

RAM : 8.00 GB

Figure 7.1: PFAS simulator system configuration

As outlined in section 4.1.2 of Chapter 4, the core component of the PFAS is the OTT authentication module, which is based on the OTT algorithm. So, it is highly essential to implement this core module. Since, this module is needed for both POS and BAS of the PFAS; the necessary scripts are identified and developed for this purpose. The scripts implemented are presented under Tables 7.1, 7.2 and 7.3 of the appendix.

Vahid. K. Alilou, a researcher at the University of Semnan in Iran, consolidated the works of various well-known researchers in the field of fingerprint authentication [137]. He implemented a simulator in Matlab for the testing and evaluation of the Conventional Fingerprint Authentication System (CFAS). This simulator mimics the normal or conventional fingerprint authentication procedure, where the original, enrolled fingerprints are stored in the FVC2002 database of the BAS. During the authentication phase, the query fingerprints are compared against the enrolled fingerprints in the BAS, and the authentication is granted or denied based on the matching score.

The feature extraction and the matching module of the CFAS were customised and integrated into OTT module of the PFAS simulator developed as part of this research. The FVC2002 database was also interfaced with the OTT module of the BAS for simulation and testing. More details on the FVC 2002 database are provided in the next section. The FAS simulator implemented by the researcher, Vahid. K. Alilou from here on will be known as CFAS.

The next section presents the test strategy.

7.2 Test strategy

The PFAS was benchmarked against the CFAS by testing against the finger template images of the FVC2002 Database. FVC2002 stands for the Second Fingerprint Verification Competition (FVC). The FVC focusses on the fingerprint verification software and algorithm assessments, thereby disseminating information on state-of-the-art developments in the domain of fingerprint technology. The FVC was held every second year between 2000 and 2006. The reason for choosing the FVC2002 database for this research is its worldwide acceptance as a benchmark dataset for evaluating the

performance of the fingerprint authentication algorithms [138]. The FVC 2002 DB1_B database is freely available for both academia and industry, and is used to test the PFAS. A subset of this database is captured as a screenshot in Figure 7.2 of the appendix. The database consists of 80 fingerprint images obtained from 10 subjects. Each subject has 8 images and the images are in TIF[7] format. The images consist of 560x296 pixels, with a resolution of 569 dpi[8] [139]. The FVC2002-DB1_B database is henceforth referred to as test database. The objective of the test strategy is to evaluate PFAS in relation to CFAS using the test database. The relevant test cases that are aimed for this purpose are explained in the next subsection.

7.2.1 Test cases

Three test cases or scenarios were identified for thoroughly testing the PFAS and CFAS. The test cases described below were used as a common criterion in the FVC 2002 competition to benchmark the performance of the FAS [138]. The aim of these test cases is to generate various biometric error rates under various test scenarios.

7.2.1.1 Test case 1

In this test case, each finger template of each test subject is compared to the remaining templates of the same subject (i.e. the ability to positively match the templates of the same individual is ascertained). The test case makes use of a positive testing approach; the test steps are presented in what follows.

1. Each finger template of each subject is compared or matched against his or her remaining 7 finger templates, but with the same pair of finger templates only compared once. This implies that if *Template1* and *Template2* of *subject1* were already compared, then *Template2* will not be compared again with *Template1*. This corresponds to 28 (i.e. 1*8*7/2) comparisons per subject and 280 (1*8*7/2) comparisons in total for all 10 subjects.
2. The similarity score or threshold *t* is based on a matching score of 0.48, as this is the threshold score calibrated in the CFAS simulator. During the calibration phase,

[7] Tag image bitmap file (TIF) is an image file format for high-quality graphics. TIF files are also called TIFF, which is an acronym for Tagged *Image Format File*.

[8] A term dpi, or DPI, is an acronym for Dots Per Inch. DPI denotes the number of pixels that appears within the span of 1 inch.

CFAS was tested with all values from 0.01 to 0.99 with increments of 0.01 to obtain an ideal EER [137]. The EER thus obtained was at a threshold of 0.48. The EER will be explained in section 7.3. It is therefore mandatory to use the same matching score in the PFAS simulator in order perform the comparison with the CFAS. A threshold *t* equal to or above 0.48 implies a correct match, whereas a threshold *t* below 0.48 implies an incorrect match.

3. The test results are recorded for all 10 subjects.
4. The expected test result for this test case is all genuine matches and allowing the authentication of users.

7.2.1.2 Test case 2

In this test case, each finger template of each test subject is compared to the corresponding templates of the remaining test subjects (i.e. the likelihood of falsely or incorrectly matching against the templates of different individuals is ascertained). The test case makes use of a negative testing approach; the steps are presented in what follows.

1. The first finger template of each subject is compared against the first finger of the remaining subjects. For example, *template1* of *subject1* is compared against the *template1* of *subject2* to *subject10*. Thus, for 10 subjects, there is 45 (10*9/2) comparisons in total.
2. The default threshold is based on the same matching score of 0.48, as in test case 1.
3. The test results are recorded for each of the 10 subjects.
4. The expected test result for this test case is all incorrect matches and thereby denying the authentication of imposters.

7.2.1.3 Test case 3

This test case follows a regressive testing approach; the steps are presented in what follows.

1. Each finger template is tested against all 80 finger templates in the database.
2. Step 1 is performed for threshold values in the range of 0.01 to 0.99 with increments of 0.01. In other words, the tests are performed using values of 0.01, 0.02, 0.03, and so on, up to a final threshold value of 0.99. This implies that there

is a total of 100 threshold values (or iterations) to test each of the 80 fingerprint templates.

3. Both the FAR and FRR (as set out in section 7.3) for each threshold value and for each finger template, are recoded in the FAR and FRR tables. The FAR and FRR consists of 100 rows and 80 columns. The Matlab script for generating the FAR and FRR tables is illustrated in Table 7.3 of the appendix.
4. Graphs are plotted for the threshold or similarity score, as a function of the FAR and FRR error rates obtained in step 3.
5. The EER, the ZeroFAR, and the ZeroFRR are calculated from the graphs obtained in step 4. The EER, ZeroFAR and ZeroFRR are explained in section 7.3.

The above test cases were executed in the CFAS and the PFAS simulator to capture the necessary test results, which are discussed in the next subsection.

7.2.2 Test results

Table 7.4 contains the full test log obtained on executing test case 1 in the CFAS and the PFAS. Figures 7.3 and 7.4 illustrate a snapshot of an instance while running this test case. It was captured during the testing of *finger template1* of *subject1* against the *finger template4* of *subject1*. The full test log that was obtained while executing test case 2 in the CFAS and the PFAS is captured in Table 7.5 in the appendix. Figures 7.5 and 7.6 illustrate a snapshot of a finger template comparison obtained while running this test case, where the *finger template8* of *subject1* was tested against the *finger template1* of *subject10*. The minutiae points marked in blue represent those minutiae belonging to the enrolled finger template, whereas minutiae points marked in red represent the minutiae belonging to the query finger template, as obtained during the authentication or matching phase.

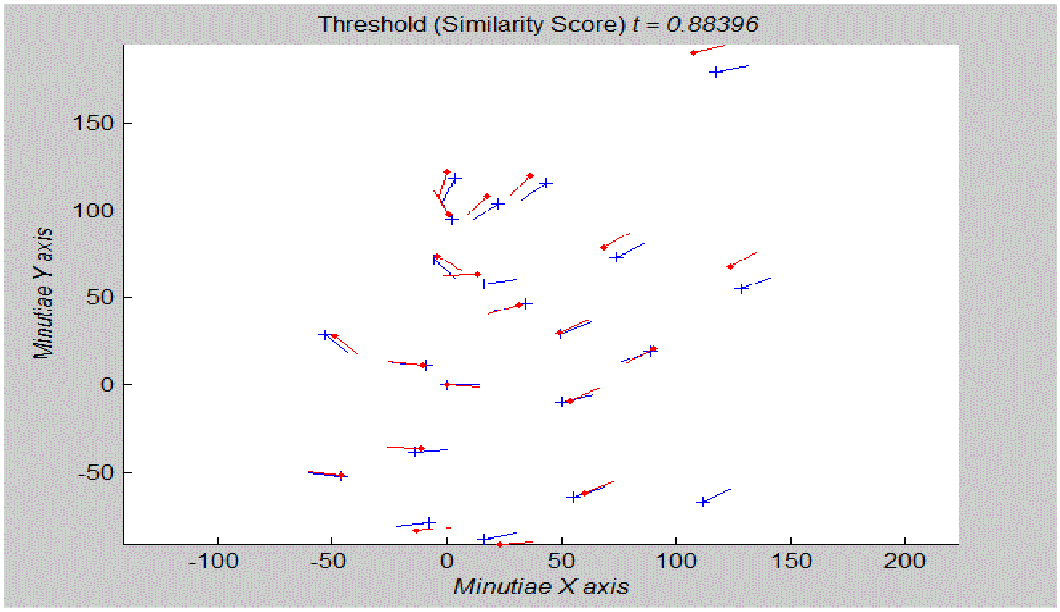


Figure 7.3: Threshold in CFAS for different finger templates of the same subject

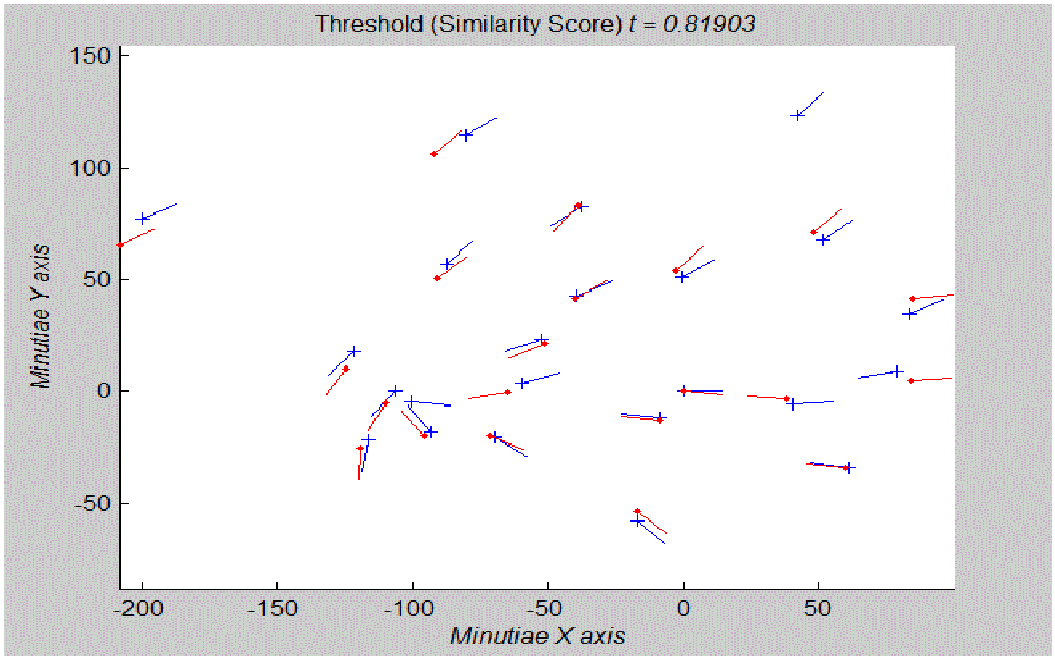


Figure 7.4: Threshold in PFAS for different finger templates of the same subject

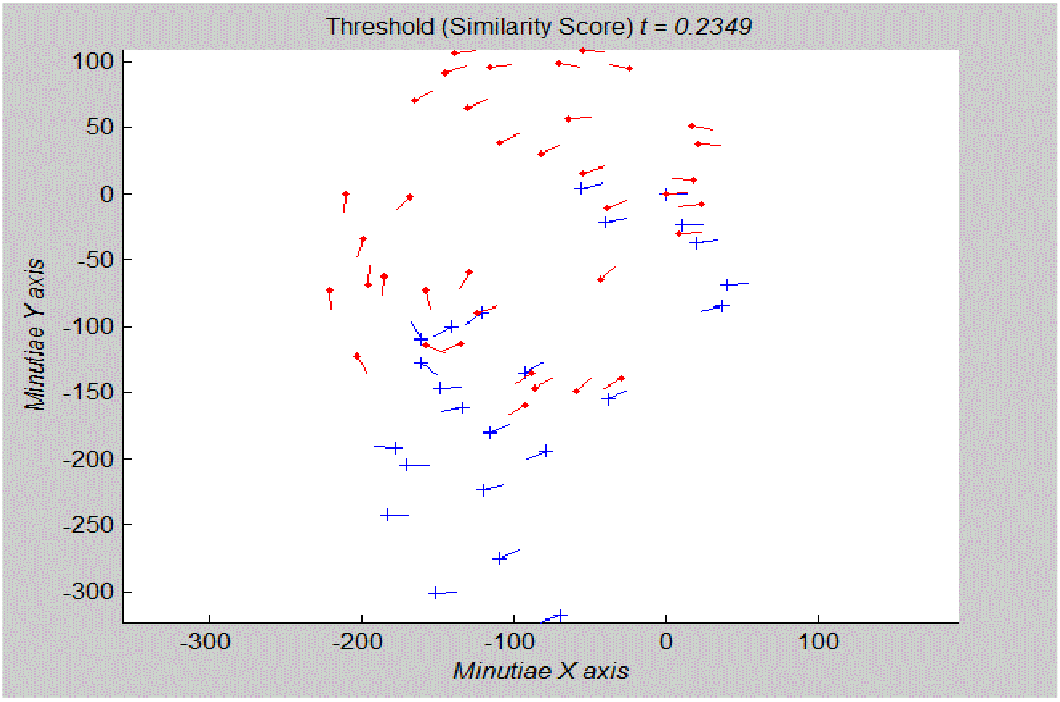


Figure 7.5: Threshold in CFAS for finger templates of different subjects

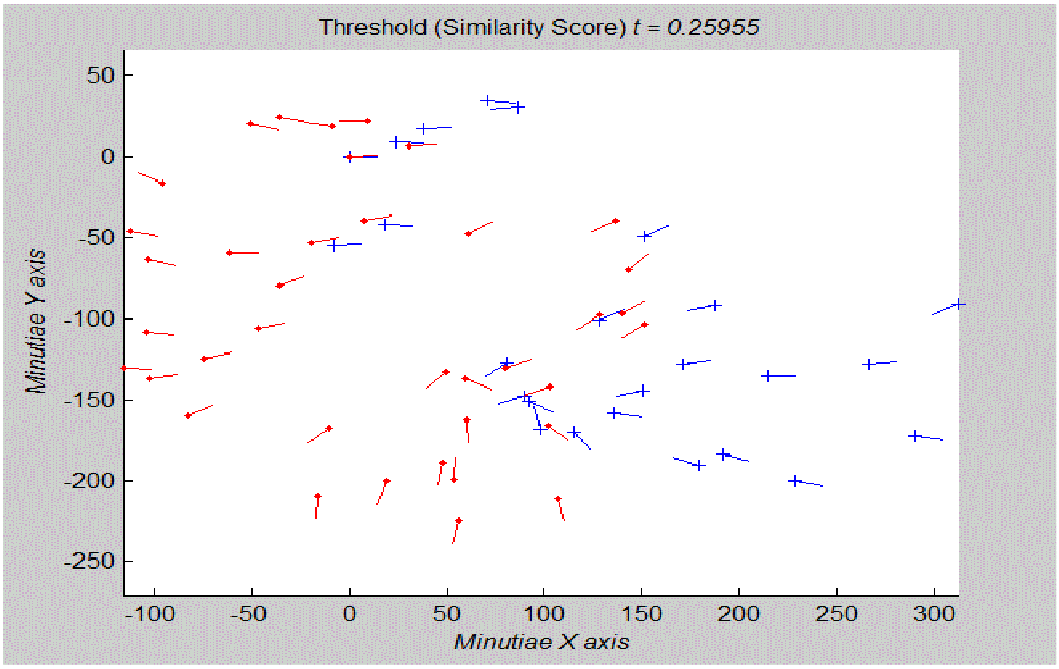


Figure 7.6: Threshold in PFAS for finger templates of different subjects

The following figures (Figure 7.7 and Figure 7.8) depict the graphs plotted between the *threshold t* and the *FAR/FRR Error Rate* of the CFAS and the PFAS, respectively. The graphs were plotted using the test results of test case 3, as captured in the FAR and FRR tables of the CFAS and the PFAS. Figures 7.9 and 7.10 in the appendix illustrate a subset of the FAR and FRR tables generated for the evaluation of the CFAS and the PFAS.

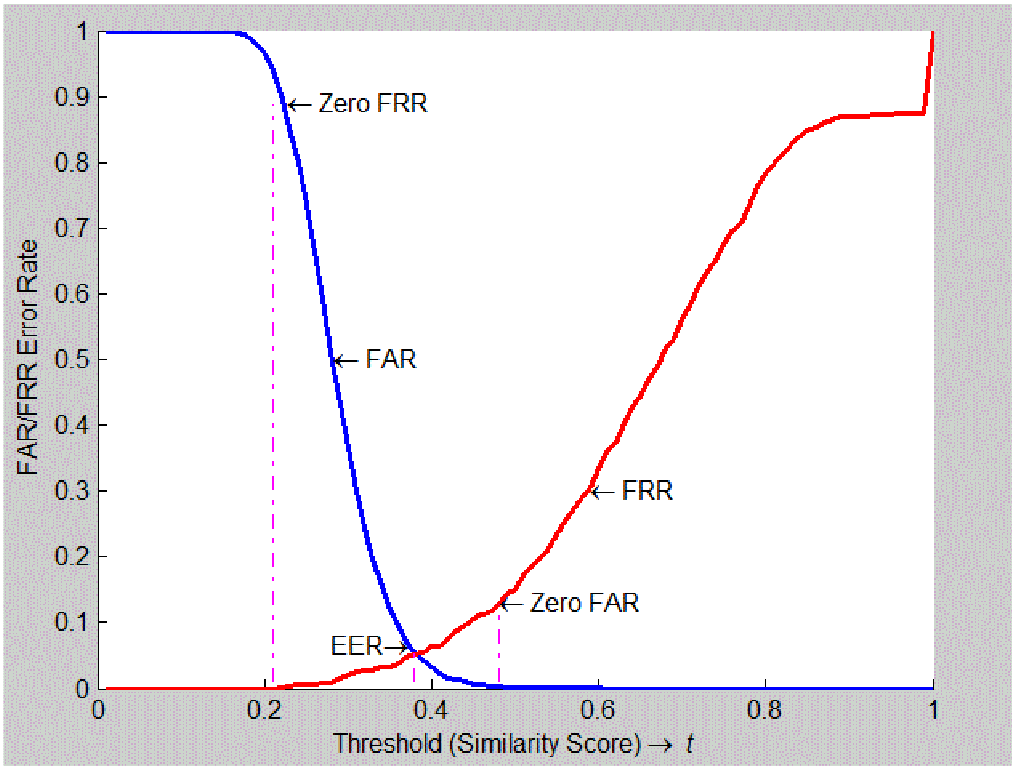


Figure 7.7: FAR/FRR against different thresholds in the CFAS

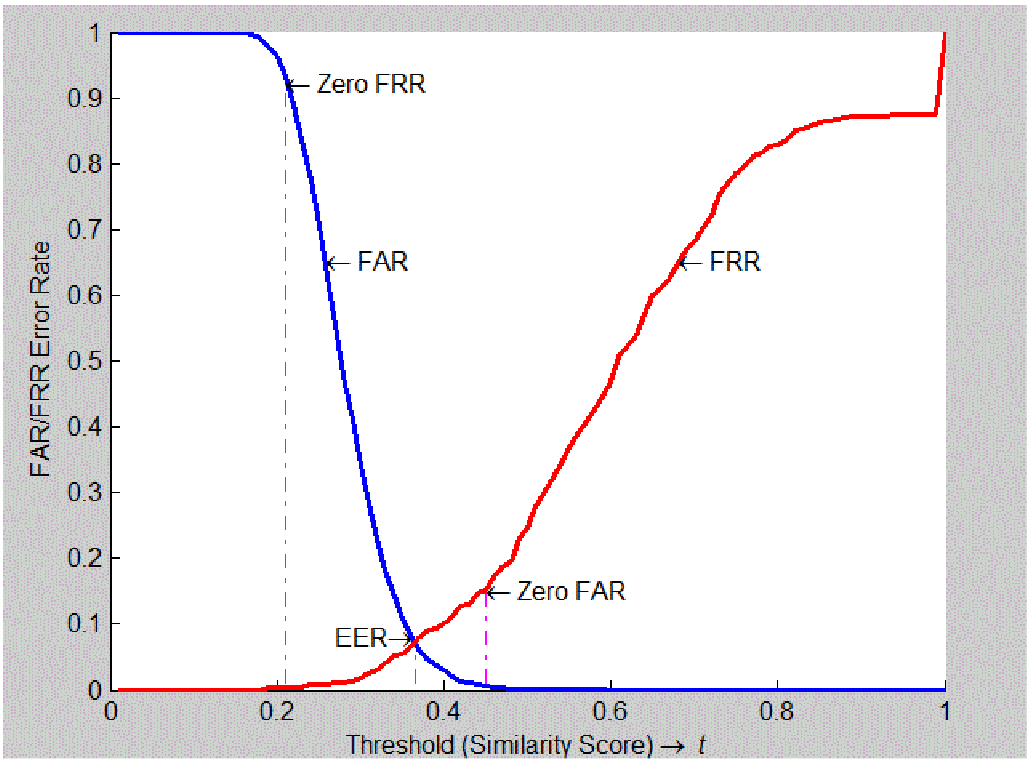


Figure 7.8: FAR/FRR against different thresholds in the PFAS

The next section focuses on the evaluation of the PFAS.

7.3 Evaluation

The aim of this section is to set out an evaluative study of the PFAS. It is essential to identify the value added of the PFAS and to bring out its uniqueness in addressing the research problem. The PFAS with respect to the CFAS is evaluated based on the test cases generated in the previous section. This section is divided into two subsections and they are as follows:

7.3.1 Performance considerations

This section focuses on the performance considerations of the PFAS, in comparison with the CFAS. Table 7.6 shows a snapshot obtained, while executing test case 1. In this case, the *finger template1* of *subject1* is matched against the *finger templates2* to *8* of *subject1*. The full test log of test case 1 is provided in Table 7.4 in the appendix.

Table 7.6: Snapshot of Test case 1: Subject1

Fingerprint Template1	**Fingerprint Template2**	**CFAS threshold *t***	**PFAS threshold *t***
1	2	0.77067	0.7298
1	3	0.68476	0.65275
1	4	0.88396	0.81903
1	5	0.6742	0.62869
1	6	0.71263	0.5808
1	7	0.5766	0.59079
1	8	0.73983	0.66725

As outlined, in test case 1 of the previous section, Table 7.6 shows a subset of the threshold *t* obtained in the CFAS and the PFAS, out of the total 280 matches. Table 7.7 shows the overall results obtained in the CFAS and the PFAS.

Table 7.7: Test case 1: Final Result

CFAS		**PFAS**	
Correct Matches	Incorrect Matches	Correct Matches	Incorrect Matches
240	40	217	63

In test case1, the expected test result was all correct matches. The CFAS obtained 83.33% correct match with a 16.67% incorrect match, while the PFAS obtained 77.5% correct match with a 22.5% incorrect match. The performance of the PFAS when compared with the CFAS, is lower by a margin of 5.83%. This drop in performance is tolerable, considering the fact that the PFAS scheme is superior to the CFAS in terms of fulfilling its key security objectives, which is due to the usage of the OTT during the authentication phase.

Table 7.8 shows a subset of the threshold *t* obtained in the CFAS and the PFAS, out of the total 45 comparisons (refer to test case 2 in the previous section). Table 7.9 contains the final results obtained in the CFAS and the PFAS.

Table 7.8: Snapshot of Test case 2: subject1 matched against other subjects

Fingerprint Template1	**Fingerprint Template2**	**CFAS threshold *t***	**PFAS threshold *t***
subject1	subject2	0.19739	0.22259
subject1	subject3	0.18019	0.21147
subject1	subject4	0.24772	0.26698
subject1	subject5	0.19508	0.22004
subject1	subject6	0.29854	0.34699
subject1	subject7	0.30151	0.21779
subject1	subject8	0.28651	0.30569
subject1	subject9	0.24175	0.27584
subject1	subject10	0.2103	0.23678

Test case 2 measured the tendency of the PFAS and the CFAS to incorrectly match a finger template of one individual against the finger template of a different individual. The PFAS and the CFAS obtained 0% incorrect match and 100% correct match out of a total of 45 comparisons performed. From Table 7.9, it is ascertained that both the PFAS and the CFAS are successful in meeting the expected test result.

Table 7.9: Test case 2: Final Result

CFAS		**PFAS**	
Correct Matches	Incorrect Matches	Correct Matches	Incorrect Matches
45	0	45	0

In test case 2, the expected test result was all incorrect matches. The PFAS and the CFAS obtained 100% incorrect match and 0% correct match out of a total of 45 matchings performed. From Table 7.9, it is ascertained that the PFAS is successful in meeting the expected test result as with the CFAS.

The equations necessary to calculate the PFAS and the CFAS error rates, as stated in test case 3 in the previous section, are consolidated in Figure 7.11.

$$\text{FAR} = \text{Falsely accepted individuals} \;/\; \text{Total number of wrong matchings'} \quad (3)$$

$$\text{FRR} = \text{Falsely rejected individuals} \;/\; \text{Total number of correct matchings'} \quad (4)$$

$$\text{EER} \rightarrow \text{FAR}(t) = \text{FRR}(t) \quad (5)$$

$$\text{ZeroFAR} = \min\{\text{FRR}(t) \mid \text{FAR}(t) = 0\} \quad (6)$$

$$\text{ZeroFRR} = \min\{\text{FAR}(t) \mid \text{FRR}(t) = 0\} \quad (7)$$

Figure 7.11: Equations to benchmark different error rates [140]

The FAR and FRR were discussed in section 2.5.2.2 of Chapter 2. To reiterate, FAR is defined as the probability that an intruder is authenticated by the system as a legitimate user, whereas the FRR is defined as the probability that an authorised person is not authenticated as a legitimate user by the system. The phrase '*Total number of wrong matchings*' in equation (3) implies the total number of incorrect matchings' that are possible and it represents the finger templates that are not belonging to a valid or a genuine user. The phrase '*Total number of correct matchings'* in equation (4) implies that the total number of correct matchings' that are possible and it denotes the finger templates of valid or genuine users.

Equal Error Rate (EER) in equation (5) is defined as the threshold *t* where the FAR equals the FRR. The EER is easy to understand and has found wide acceptance as a single measure of system performance (a performance comparison between multiple systems is simplified if the performance is expressed using a single metric) [140]. ZeroFAR in equation (6) is defined as the lowest FRR at which no false acceptances occur, whereas ZeroFRR in equation (7) is defined as the lowest FAR at which no false rejections occur. The CFAS and the PFAS were compared qualitatively based on the set of key objectives

that were proposed in this research, as well as the final test results obtained from the equations listed in Figure 7.9. A comparison is drawn between the PFAS and the CFAS in Table 7.10.

Table 7.10: Comparison of the PFAS and the CFAS

FAS	PS	MA	RC	RS	RRA	ZeroFAR	ZeroFRR	EER
CFAS	*W*	*W*	*W*	*N*	*W*	*at* FRR = 0.13	*at* FAR = 0.89	0.068 at $t = 0.38$
PFAS	*S*	*S*	*C*	*S*	*S*	*at* FRR = 0.15	*at* FAR = 0.92	0.07 at $t = 0.365$

The abbreviations used in Table 7.10 are listed in what follows.

FAS– Fingerprint Authentication System

PS- Privacy and security

MA- Mutual authentication

RC- Resilience to compromise of finger template

RS- Revocation support

RRA- Resilience to replay attacks

ZeroFAR- Zero False Acceptance Rate

ZeroFRR- Zero False Rejection Rate

EER- Equal Error Rate

CFAS- Conventional Fingerprint Authentication System

PFAS- Proposed Fingerprint Authentication System

N- Nil

W- Weak

C- Comparable

S- Strong

The comparison of the security properties of the CFAS and PFAS, as provided in Table 7.10, are based on a quanlitative scale. The quanlitative scale denotes various levels of security, which are classified as either *N-Nil*, *W-Weak*, *C-Comparable*, or *S-Strong*. The security level *N* implies that the corresponding security property is absent, whereas *W* signifies the weakest security level, *S* the strongest security level, and *C* a security level between *W* and *S*.

As illustrated in Table 7.10, the *privacy and security (PS*) property is classified as weak in the CFAS, and strong in the PFAS. In the CFAS, the original finger templates are captured and stored in the BAS during enrollment. This poses a significant security risk as all finger templates will be permanently lost if the finger template database is compromised. In the PFAS, the original finger templates are transformed into a series of IOTTs as explained in Chapter 4, and the original finger template is removed after the IOTT generation process. As a result, if the database is compromised, the attacker will not be able to reverse engineer the finger templates in its original form. Furthermore, in the PFAS, all communication between the POS and BAS are encrypted. As a result, the privacy and security provided by the PFAS is considered strong, whereas it is considered to be weak in the CFAS.

The weak privacy and security in the CFAS negatively impacts the integrity of mutual authentication. This is due to the fact that, in the CFAS, there is a possibility that authentication can be successfully carried out using a compromised finger template. Therefore, the ability of the CFAS to perform *mutual authentication (MA)* is not as strong as in PFAS. In the PFAS, the mutual authentication is highly dependable due to the strong *privacy and security* afforded by the system.

The *resilience to compromise of finger template (RC)* property in the CFAS is classified as weak, as the original finger templates are stored in the database. Moreover, due to weak *privacy and security*, the finger templates are susceptible to compromise in the CFAS. In the PFAS, the original finger templates are never stored in the database and hence, the *RC* property is classified as comparable. Even if the finger template is compromised in the PFAS, the authentication will not succeed as the generated OTT is unique for each authentication session.

In the PFAS, the *revocation support (RS)* property is classified as strong in PFAS, owing to the fact that even if the template is compromised, it is still revocable. The original finger template is intentionally distorted to an IOTT during the enrollment phase and stored in the BAS. In the event of a compromise, a new transformed finger template can be easily generated. In the CFAS, the support for revocation is absent. The compromise of a finger template in the CFAS implies that the original finger template is lost forever.

The *resilience to replay attacks (RRA)* property is classified as strong in PFAS, as the attempted usage of a compromised OTT in a replay attack will not succeed. This is due to the fact that, the OTT is unpredictable and distinct in each authentication session. This does not hold for the CFAS, as the finger templates are identical for every authentication session, and hence predictable. Thus, the vigilant reuse of an old instance of a finger template to launch a replay attack is likely to succeed in the CFAS. Therefore, the *RRA* property is classified as weak in the CFAS.

It is observed from Table 7.10 that the ZeroFAR of the CFAS is at FRR 0.13. This means that the CFAS will not be allowing the authentication of any imposters, when it has an FRR of 13% (an error rate of not allowing the authentication of 13% of genuine users). In the case of the PFAS, the ZeroFAR is at FRR 0.15. It implies that the PFAS will not be allowing the authentication of any imposters, when it has an FRR of 15%. In other words, an FRR of 15% in PFAS indicates that it will be rejecting the authentication of 15% of genuine users when not allowing the authentication of any imposters.

The ZeroFRR of the CFAS is at FAR 0.89. The ZeroFRR is the FAR value that is needed for not denying access to any valid or genuine users. This means that CFAS will not be rejecting the authentication of any genuine users, when it has an FAR of 89%. An FAR of 89% in CFAS implies that it will allow the authentication of 89% of imposters. In the case of PFAS, the ZeroFRR is at FRR 0.92. This implies that the PFAS will not be rejecting the authentication of any genuine users, when it has an FAR of 92%. In other words, an FAR of 92% in PFAS indicates that it will allow the authentication of 92% of imposters when not rejecting the authentication of any genuine users.

The EER of the CFAS is at 0.068 when the threshold t is at 0.38 – that is, the FAR and FRR of the CFAS is equal at this threshold value. In the case of the PFAS, the EER is at 0.07 when the threshold t is at 0.365. It is acceptable to keep the same threshold in the PFAS as in the CFAS, as the EER varies by a margin of only 0.002 between the two systems.

From the analysis conducted in this section, it is observed that the EER of the PFAS and the CFAS differ by a margin of only 0.002. The FRR of the PFAS is higher than the FRR of the CFAS by a margin of 2% when achieving ZeroFAR. Similarly, the FAR of the PFAS is higher than the FAR of the CFAS by a margin of 3% when achieving ZeroFRR.

The marginal variations in the ERR, ZeroFAR, and ZeroFRR error rates of the PFAS are tolerable considering the enhanced security that the PFAS offers when compared with the CFAS. *In summary, after taking all these factors into consideration, the PFAS is comparable in its performance with the CFAS and can be deployed in a POS transaction environment. Moreover, the PFAS offers security that is superior to that of the CFAS, in its attempt to achieve the research goal, which is to incorporate a robust fingerprint biometric authentication mechanism in POS terminals to authenticate transactions performed using magnetic stripe bank cards.*

The next subsection focuses on the usability aspects of the PFAS.

7.3.2 Usability aspects

After analysing the results obtained in the previous section, it is established that the usage of the PFAS is feasible. Furthermore, the PFAS has comparable performance to the CFAS and offers a higher security in its aim to address the research problem. The following paragraphs will shed light on improving the usability aspects of the PFAS and biometric authentication systems in general.

Fingerprint authentication is still a challenging problem for reliable personal authentication because of the complex distortions involved in two impressions of the same finger. Since the vast majority of fingerprint matching algorithms rely on minutiae matching, minutiae information is regarded as a highly significant feature for the FAS. The accuracy of any FAS depends on the image quality, image enhancement methods, feature extraction algorithms, and feature set pre-processing/post-processing algorithms. Whilst there are certain notable opportunities with the implementation of fingerprint or biometric authentication systems in general, a thorough approach is needed in the implementations of the biometric authentication systems in order to derive the inherent benefit. Though, biometric authentication has been documented to reduce financial fraud as compared to knowledge-based mechanisms, there is an inherent uncertainty and risk of error that can be associated with its probabilistic nature. Hence, a systematic methodology is needed to warrant that biometric systems perform to the expected levels to ensure that the uncertainties and risks are mitigated.

It is a well-known fact that the FAR and the FRR is a common problem associated with any biometric modality as it involves human and biological characteristics [141, 142]. In a practical scenario, a low FAR and a high FRR would ensure that any imposter will not be allowed access. It would also mean that a genuine user may need to present the fingerprint multiple times before authorising access. The FAR and FRR are closely related to each other and if one decreases, the other increases and vice versa. During the enrollment phase, it is a good practice for all users to present their finger on the fingerprint scanner in such a way that it captures the minutiae from the middle of the fingerprint. This will ensure uniformity and constancy of the extracted features in the POS and the BAS. It is also advisable to acquire multiple images of the same fingerprint during the enrollment and the verification stage. Further, the best image generated must be presented as input to the OTT module. This will further normalise the FAR/FRR during authentication.

As the PFAS is aimed for a highly sensitive POS transaction framework, the users' fingerprint must be captured accurately to minimise the FAR/FRR rates. The proposed research is laid out in a financial transaction environment. Hence, it is not advisable to authenticate a user based on an incorrect OTT generated at the BAS. Thus, the FAR/FRR must be kept very minimal, as the PFAS considers that the security of the user and authenticity of the transactions weighs more than the FAR/FRR caused by inaccurate fingerprint captures.

The usability aspects of the PFAS are further analysed based on different transaction scenarios which are explained under the following subsections.

7.3.2.1 Genuine transaction scenario

In this scenario, the magnetic stripe bank card belonging to the legitimate owner is used to conduct a genuine transaction. In the PFAS, the finger templates of the original cardholder are already enrolled at the start of the transaction, as explained in section 4.1.2 of Chapter 4. When the card is presented for payment at the POS terminal, the user is asked to present the finger template. The POS terminal and the BAS communicate according to the PFAS protocol, which proceeds to the user logon phase, followed by the fingerprint authentication phase. In this phase, the OTT generated at the POS for the current transaction session is compared against the OTT derived at the BAS, and a degree of similarity is calculated. The OTT generated at the POS is based on the original finger

template of the cardholder and hence it matches the OTT generated at the BAS. As a result, the transaction is authorised and proceeds as normal in the POS terminal.

7.3.2.2 Illegal or fraudulent transaction scenario

In this scenario, a cloned magnetic stripe bank card belonging to an attacker is presented at the POS terminal for conducting an illegal or fraudulent transaction. Here, the finger template presented during the transaction belongs to the attacker. The OTT generated at the POS is based on the attacker's finger template, whereas the OTT generated at the BAS is based on the finger template of the original cardholder. Hence, the comparison of the OTTs fails during the authentication phase, and the transaction is not authorised to proceed further at the POS terminal.

The next section summarises the current chapter.

7.4 Summary

The main focus of this chapter was to conduct a thorough evaluation of the proposed FAS (PFAS) against the conventional FAS (CFAS) to ascertain the value add that it offers in resolving the research problem. For this purpose, the PFAS implementation has been carried out in Matlab. The test strategies were formulated and the relevant test cases were designed accordingly to test the PFAS. This was followed by the actual testing and the capturing of the test results. The test results obtained were then mapped to relevant tables and graphs. Further, a thorough evaluation was conducted by analysing the test results to establish the performance considerations of the PFAS against the CFAS. From the analysis of the test results, it was learned that the performance of the PFAS is on par with the CFAS with the additional benefit of offering a superior security than the CFAS. This chapter also studied the usability aspects of the PFAS with respect to different transaction scenarios. The next chapter aims to conclude the current study.

8
Conclusion

The usage of magnetic stripe bank cards in POS terminals has become ubiquitous and revolutionised the modern world. It is acknowledged globally as the most convenient and the principal electronic payment scheme. However, it has been demonstrated that the inherent security vulnerabilities of magnetic stripe bank cards can be readily exploited, and hence these cards form an attractive target for criminals. The usage of the cloned magnetic stripe bank cards in POS terminals has grown significantly over the years. Although the introduction of the EMV card (chip and pin technology) has given the impression that the card cloning issue in POS terminals has been alleviated to some extent, in reality, this is not the case. This is due to two major reasons. Firstly, card cloning is a prevalent crime committed in non-EMV compliant countries such as Australia, New Zealand and partially EMV compliant countries such as the U.S., UK, Scotland, Ukraine, Italy and India, amongst other countries. Secondly, criminals clone the EMV card and damage the chip, which causes the card to revert to the magnetic stripe; this imples that all subsequent transactions are executed as normal magnetic stripe transactions at the POS terminal. Billions of dollars are lost and are written off by banks every year due to fraudulent transactions performed using cloned cards in POS terminals. This problem was addressed in this research by the development of a robust FAS, which authenticates the transactions performed using magnetic stripe bank cards at POS terminals. The current chapter is structured as follows. Sections 8.1 and 8.2 present the research synopsis and discuss the significance of this research. Section 8.3 establishes the specific contribution of this research. In section 8.4, avenues for future research are provided, and limitations of the research are considered. In section 8.5, the possibilities and use cases of this research are discussed.

8.1 Research synopsis

This study began with the analysis of the card cloning concept in POS terminals using the magnetic stripe bank cards and subsequently arrived at the motivation and problem

statement of the research. An outline of how to tackle the research problem was then considered. It was learned that although existing security implementations are in place to authenticate magnetic stripe bank cards at POS terminals, they have limitations and can lead to serious security vulnerabilities. This emphasised the need of biometrics in the form of an FAS in the POS transaction framework to address the research problem.

As this study progressed further, it was discovered that although the usage of an FAS in its present form can authenticate magnetic stripe bank card transactions at POS terminals, it may lead to critical security issues. This is owing to the fact that biometric systems have inherent security vulnerabilities, and FASs' are also prone to these security vulnerabilities. This called for the need of a robust FAS, which can address the existing biometric specific vulnerabilities to some extent and at the same time can solve the research problem. The research further proposed a novel FAS (termed PFAS) to *authenticate transactions performed using magnetic stripe bank cards at POS terminals*. The research then provided a detailed design of the FAS and the underlying FAS protocol with an analysis of different modules, the interfacing, and communication between them. The Proposed Fingerprint Authentication System (PFAS) was modelled using ProVerif model and the verification and validation were conducted using the ProVerif scripts. This was followed by an evaluation of the PFAS by simulating appropriate test scenarios in Matlab. Further, the performance considerations, usability aspects, and critique of the PFAS were conducted with respect to the CFAS.

8.2 Significance of this research

In this research study, a thorough analysis of different biometric schemes was conducted, and it was learned that the existing biometric schemes have serious security issues, and the FAS is also not free from these vulnerabilities. The research took a turning point at this step and the thought from there on was to build a FAS that can alleviate the inherent biometric vulnerabilities associated with FAS to some extent and at the same time can resolve the research problem.

Different finger template protection schemes were analysed for this study, and it was found that existing schemes do not meet their expected security objectives or properties, which is the root cause of current biometric vulnerabilities. Hence, the proposal for a

robust FAS scheme was made and with a vision of meeting the objectives that the existing schemes failed to meet. To achieve the objectives of the PFAS, this research conceptualised a security model that derives an unpredictable, one-time finger template by inheriting the principles of Biohashing and the OTP scheme. During the authentication phase, OTT derived for authenticating the user is matched against the OTT at the BAS and a decision is taken accordingly. *The core idea of the PFAS is that the original finger template of the user is never stored in the POS or in the BAS, and that the template is never transmitted during authentication, in its aim to provide security that is superior to that of existing FAS schemes.*

The verification, validation, and evaluation of the PFAS revealed that it fulfills all its objectives and that it achieves a practically good performance. The most important aspect of this research is that, in its endeavour to address the research problem and to achieve the research goal, it could also propose a solution to alleviate a subset of the key security vulnerabilities associated with biometrics. Furthermore, it was established that the framework is effective and distinctive in its approach towards the authentication of magnetic stripe bank card transactions at POS terminals. The next section presents the specific contributions of this work.

8.3 Specific contributions of this research

In the conventional FAS (CFAS), it is a requirement to store the original finger template of the user in the BAS database. This opens the door for serious security vulnerabilities as the compromise of the biometrics is of permanent nature. Furthermore, this can lead to cross platform application attacks using the compromised finger templates from the BAS. This research abstracted the idea of using a unique finger template in the form of an OTT to make the authentication process unique during each transaction session. The PFAS is distinctive in the sense that the original finger template of the users will not be stored in the BAS database and even if the database is compromised, it is impossible to reverse engineer the original finger template of the user. This essentially makes the spoofing attacks on the database insignificant and further eliminates the cross platform attacks.

The research conceptualises a unique OTT generation algorithm. The idea of adding date stamps and time stamps as security salts during the OTT generation brings up the

elements of individuality and unpredictability. In addition, this research also brings up the concept of logon request and logon reply, which is a prerequisite of the authentication phase.

The current research showcases an FAS framework with a superior authentication mechanism in a POS transaction environment. The proposed authentication framework binds a magnetic stripe bank card to its user biometrically and hence can prevent the use of cloned magnetic stripe bank cards at POS terminals. These contributions led to the proposal of a unique FAS to authenticate transactions performed using magnetic stripe bank cards at POS terminals without transmitting the original fingerprint templates. The next section focuses on future research and limitations.

8.4 Future research and limitations

Although this research achieved its objectives, there remains potential for improvement. The scope for improvement is elaborated in the following subsections.

8.4.1 Improvements in the enrollment process

The first improvement that could be made to the PFAS in general is to improve the enrollment process. The current enrollment process is based on the capture of a single finger template from the user. The enrollment process could be improved by capturing at least 4 finger templates. The IOTT generation and the OTT generation process in the BAS and the POS could be improved accordingly to cater for this change. The reason behind the suggestion for this enhancement is to bring in an element of randomness or unpredictability during the finger template verification. The idea is that during the authentication phase, the user can be asked to prompt for any random finger from the set of 4 finger templates. The multiple fingerprint enrollment processes will also address the *injury cases* in which the user has an option to present an alternate finger if the requested finger is injured. Although these changes will add extra overhead during the enrollment and processing at the BAS and the POS terminal, it can improve the security of the system.

8.4.2 Incorporating a cloned hot card list

Provision can be made available to add an extra message after the authentication leg

between the POS terminal and the BAS. This is to inform the BAS about the status of the authentication. If the authentication fails, the BAS can then conclude that the authentication failed for the corresponding PAN. The BAS can ascertain that the card is cloned and could mark the card as hot and add it to the cloned hot card list. The process in the POS terminal could also be modified to incorporate this change, and a cloned hot card file needs to be downloaded every morning on the terminal. So, when a user comes for the transaction; after card swipe, the PAN must be compared against the hot card list, to determine if the card is cloned or not. It should be noted that the implications of the storage requirements, connectivity, download time, and cost must be carefully studied on POS terminals and BAS before incorporating this change.

8.4.3 Increasing the coverage in addressing the FAS vulnerabilities

This research addressed a subset of the FAS vulnerabilities laid out under section 2.6 of Chapter 2. The focus was on alleviating the major vulnerabilities based on the key objectives identified under section 4.1.1 of Chapter 4. A further research can well be conducted to address the compromise of the original finger templates at the sensing device or on the channel between the sensing device (scanner) and the POS terminal. Additionally, a study can be also conducted on the vulnerabilities occurring as a result of the dishonest acts from merchants or vendors. By doing this, the coverage of addressing the FAS vulnerabilities can be increased.

8.4.4 Scalability

For simplicity and clarity purposes, the PFAS is laid out based on the key entities in the transaction framework. In the real world implementations, this may not be the case. The IOTT/OTT generation process and storage in the PFAS can also be improved as millions of users with billions of transactions occur on a daily basis. Hence, the IOTT/OTT could only be generated on demand to address the scalability constraints. The scalability aspects can be studied further, depending on the use case where the proposed work needs to be deployed.

8.4.5 Interoperability

Further study can be conducted to understand the interoperability aspects between the different fingerprint scanners. This is to understand and address cases such as, whether the finger template enrolled using the fingerprint scanner of manufacturer X will match

against the finger template presented at the fingerprint scanner of manufacturer Y, and vice versa. Means for addressing lack of interoperability form an additional avenue of research.

8.4.6 Offline POS terminals

The PFAS was designed from the perspective of online POS terminals. However, a study can be conducted to handle scenarios in which the POS terminal loses the communication with the BAS. Also a study may be conducted to analyse how the PFAS can be adapted only to offline POS terminals, which are not at all connected to the BAS.

This section has addressed some of the future research possibilities and limitations of this research. The next section looks into the cost effectiveness analysis and adaptability of the PFAS.

8.5 Cost-effectiveness analysis and adaptability

This section focuses on discussing the cost effectiveness analysis and adaptability of the PFAS in the real world. The PFAS consists of two key components. They are the POS terminal and the BAS. POS terminals with inbuilt fingerprint biometric capabilities are already available from a variety of vendors and the cost varies by a very small margin when compared to the cost of existing POS terminals that are currently used in the payment industry [6]. The BAS can be seamlessly integrated to existing banking servers, or it can be implemented as a standalone system. The cost implications of the servers are on par, as the prices of hard drives and extra memories are inexpensive these days. Moreover, no special hardware upgrades are required between the POS terminal and the banking server, during the implementation of the PFAS. The software upgrades needed to cater for the PFAS in the POS terminal and the banking server will not be a major issue, as the upgrades can be effortlessly integrated into the existing authentication flow of the transaction. In addition, the fingerprint enrollment and card issuing processes are neither expected to be time consuming nor incur significant costs, considering the processing power of the POS terminals and servers.

In 2014, the global financial losses due to card fraud were well over $12 billion, of which the losses due to card cloning alone were in the range of $2 billion [2, 20]. It is worthwhile for the payment industry to consider the implementation of the PFAS, to

address financial losses due to card cloning. The cost-effectiveness ratio of the PFAS can be estimated by comparing losses due to card cloning against the cost of implementing the PFAS. Cost-effectiveness analysis (CEA) is essentially a type of economic evaluation in which the costs of projects, procedures or programs are compared with outcomes measured in natural units [136]. It is estimated that the PFAS implementation costs $100 million and if it prevents at least 20% of financial losses due to card cloning; which will be equal to $400 million (20% of $2 billion). The cost effectiveness ratio, thus becomes 0.25 ($100 million/$400 million). This implies that if the payment industry spends 5% of the card cloning losses (5% of $2 billion) towards implementing the PFAS, then it can bring down the losses by a margin of 25%. Therefore, the important factor that the payment industry should consider is that the heavy financial losses occurring as a result of card cloning can certainly be brought down by investing a little on the PFAS. Moreover, all the stakeholders in the payment transaction chain such as the cardholders, card manufacturers, retail merchants, payment transaction network, and banking companies will certainly benefit by investing in the PFAS.

The next section concludes this research by describing its application possibilities and use cases.

8.6 Possibilities and use cases of this research

This research study finds its use in numerous use cases of which the most significant ones are included in this section. The existing scheme can be deployed by the banks across their ATMs and mobile device platforms to bring robustness on the payment using the magnetic stripe bank cards. The concept of using an OTT for authentication can be used in scenarios where existing fingerprint authentication schemes are used. The idea can also be well adapted to other biometric modalities such as the iris scans, facial scans, etc. The same concept can also be incorporated to multi-factor biometric schemes, where more than one biometric modality is required for authentication. The PFAS can be easily migrated to address the vulnerabilities inherent in smart cards. The same concept can also be easily adapted to biometric cardless payment systems, online shopping, and e-banking. Avenues for future research, as presented in section 8.4, may also be explored, as it is important to realise that no system can be completely secure, and that no single defensive mechanism will comprehensively protect a system.

Bibliography

The following are the list of references that were used in the body of the thesis.

[1] M. Salajegheh, B. Priyantha, and J. Liu., "Unleashing The Wild Card for Mobile Payment," in *Proc. of the 2014 Int. Conf. on Pervasive Computing and Communications, PerCom, 24-28 Mar., 2014, Budapest*. IEEE, 2014, pp. 121-129. [Online].Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6813952.[Accessed: Apr. 20, 2015].

[2] K. Piero and J. Finefrock, "Diebold Stops ATM Fraudsters In Their Tracks With World's Most Secure Anti-Skimming Card Reader," 2014. [Online]. Available: http://news.diebold.com/press-releases/diebold-stops-atm-fraudsters-in-their-tracks-with-worlds-most-secure-anti-skimming-card-reader.htm.[Accessed: Apr. 21, 2015].

[3] D.R. Hayes, "Skimming the Surface: How Skimmer Fraud Has Become a Global Epidemic," 2014. [Online]. Available: http://www.accaglobal.com/content/dam/acca/global/PDF-technical/other-PDFs/skimming-surface.pdf.[Accessed: Apr. 22, 2015].

[4] "Staying Ahead of ATM Skimmers," 2015. [Online]. Available: http://www.atmatom.com/staying-ahead-of-atm-skimmers/.[Accessed: Apr. 22, 2015].

[5] M. Bond, O. Choudary, S.J. Murdoch, S. Skorobogatov, and R. Anderson, "Chip and Skim: cloning EMV cards with the pre-play attack," in *Proc. of the IEEE Symposium on Security and Privacy, SP, 18-21 May, 2014, San Jose, CA,* IEEE , 2014, ISSN. 1081-6011, pp. 49-64. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6956556.[Accessed: Apr. 23, 2015].

[6] H.A. Rad, M.B. Tehrani, K. Samsudin, and A.R. Ramli, "A Simple and Highly Secure Protocol for POS Terminal," in *Proc. of the 2nd Int. Conf. on Environmental and Computer Science, ICECS, 28-30 Dec., 2009, Dubai.* IEEE , 2009, ISBN. 978-1-4244-5591-1, pp. 204-207. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp? arnumber=5383526. [Accessed: May 20, 2015].

[7] C. Li, J. Chen, and J. Luo, "Locating POS Terminals from Credit Card Transactions," in *Proc. of the IEEE International Conference on Data Mining, ICDM, 14-17 Dec., 2014, Shenzhen*. IEEE, 2014, ISSN. 1550-4786, pp. 280-289. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7023345. [Accessed: May 20, 2015].

[8] H. Guo and B. Jin, "Forensic Analysis of Skimming Devices for Credit Fraud Detection," in *Proc. of the 2nd IEEE International Conference on Information and Financial Engineering, ICIFE, 17-19 Sep., 2010, Chongqing*. IEEE, 2014, ISBN. 978-1-4244-6927-7, pp. 542-546. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5609418. [Accessed: May 20, 2015]

[9] M. Harwood, 2010. "National Retail Federation and eBay Join Forces to Thwart Organized Retail Crime," [Online]. Available: http://www.securitymanagement.com/news/ national-retail-federation-and-ebay-join-forces-thwart-organized-retail-crime-006881.[Accessed: May. 18, 2015]

[10] "Credit Card Fraud Statistics and Facts," 2009. [Online]. Available: http://www.spamlaws.com/credit-fraud-stats.html.[Accessed: Apr. 29, 2015]

[11] E. Strickland, "Blood and Money," *Spectrum, IEEE* , vol. 49, no. 6, ISSN. 0018-9235, pp. 36-41, 2012. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp? arnumber=6203965. [Accessed: May 18, 2015].

[12] "Card cloning suspects to appear," 2010. [Online]. Available: http://www.news24.com/SouthAfrica/News/Card-cloning-suspects-to-appear-20100624.[Accessed: May 21, 2015].

[13] J. Kiernan, "Credit Card & Debit Card Fraud Statistics," 2015. [Online]. Available: http://www.cardhub.com/edu/credit-debit-card-fraud-statistics. [Accessed: Apr. 20, 2015].

[14] D.R. Smith, "Chip credit-card are no safer," 2015. [Online]. Available: http://davidrs.com/wp/chip-credit-cards-are-not-safer/.[Accessed: Apr. 16, 2015].

[15] I. Sakharova, "Payment Card Fraud: Challenges and Solutions," in *Proc. of the IEEE International Conference on Intelligence and Security Informatics, ISI, 11-14 Jun., 2012*, *Arlington, VA*. IEEE, 2012, ISBN. 978-1-4673-2105-1, pp. 227-234. [Online]. Available: http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=6284315. [Accessed: Apr. 12, 2015].

[16] "Beware credit card cloning," 2014. [Online]. Available: http://www.talktalk.co.uk/money/credit-centre/credit-card-cloning.html.[Accessed: Feb. 16, 2015].

[17] H.K. Lu, A.M. Ali, S. Durand, and L. Castillo, "A New Secure Communication Framework for Smart Cards," in *Proc. of the 6th IEEE Consumer Communications and Networking Conference, CCNC, 10-13 Jan., 2009, Las Vegas, NV*. IEEE, 2009, ISBN. 978-1-4244-2309-5, pp. 1-7. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp? tp=&arnumber=4784726.[Accessed: Apr. 13, 2015].

[18] U.G. Ceipidor, C.M. Medaglia, A. Marino, S. Sposato, and A. Maroni, "KerNees A protocol for mutual authentication between NFC phones and POS terminals for secure payment transactions," in *Proc. of the 9th Int. Conf. on Information Security and Cryptology, ISCISC, 13-14 Sep., 2012, Tabriz*. IEEE, 2012, ISBN. 978-1-4673-2387-1, pp. 1-7. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6408203.[Accessed: Apr. 24, 2015].

[19] J. Kiernan, "Credit Card & Debit Card Fraud Statistics," 2015. [Online]. Available: http://www.cardhub.com/edu/credit-debit-card-fraud-statistics. [Accessed: Apr. 24, 2015].

[20] M.K. Mishra and R. Dash, "A Comparative Study of Chebyshev Functional Link Artificial Neural Network, Multi-Layer Perceptron and Decision Tree for Credit Card Fraud Detection," in *Proc. of the Int. Conf. on Information Technology, ICIT, 22-24 Dec., 2014*, *Bhubaneswar*. IEEE, 2014, ISBN. 978-1-4799-8083-3, pp. 228-233. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7033327. [Accessed: Jun. 11, 2015].

[21] E. Duman, A. Buyukkaya, and I. Elikucuk, "A Novel and Successful Credit Card Fraud Detection System Implemented in a Turkish Bank," in *Proc. of the 13th Int. Conf. on Data Mining Workshops, ICDMW, 7-10 Dec., 2013, Dallas, Texas*. IEEE, 2013, ISBN. 978-1-4799-3143-9, pp. 162-171. [Online]. Available: http://ieeexplore. ieee.org/xpl/articleDetails.jsp?arnumber=6753916. [Accessed: Jun. 11, 2015].

[22] "Banks reluctant to pay victims of chip-and-PIN fraud," 2010. [Online] Available: http://www.timesonline.co.uk/tol/money/consumer_affairs/article5575295.ece. [Accessed: Jun. 12, 2015].

[23] "Credit Card Cloning," 2014. [Online]. Available: www.bbc.co.uk/insideout/east/series3/credit_card_cloning.shtml. [Accessed: Apr. 16, 2015].

[24] "Millions in danger from chip and pin fraudsters," 2006. [Online]. Available: http://www.dailymail.co.uk/news/article-389084/Millions-danger-chip-pin-fraudsters.html.[Accessed: Jun. 13, 2015].

[25] V. Piuri and F. Scotti, "Biometrics Privacy: Technologies and Applications," in *Proc. of the 2011 Int. Conf. on Security and Cryptography, SECRYPT*, *18-21 Jul., 2011, Seville*. IEEE, 2011, pp. 1-7. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6723863. [Accessed: Jun. 13, 2015]

[26] M. Fons, F. Fons, E. Canto, and M. Lopez, "Hardware-Software Co-design of a Fingerprint Matcher on Card," in *Proc. of the IEEE Int. Conf. on Electro/information Technology, FITME, 7-10 May, 2006, East Lansing, MI*. IEEE, 2006, ISBN. 0-7803-9593-X, pp. 113-118. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4017681.[Accessed: Jun. 23, 2015]

[27] R.A. Kharade and M.S. Khumbar, "An Identity-Authentication System Using Fingerprints," 2012, International Journal of Engineering Research and Applications (IJERA), [Online]. Available: http://www.ijera.com/papers/Vol2_issue6/AV26303311.pdf.[Accessed: Jun. 23, 2015].

[28] M. Iqbal and M. Rizwan, "Application of 80/20 Rule in Software Engineering Waterfall Model," in *Proc. of the Int. Conf. on Information and Communication Technologies, ICICT, 15-16 Aug., 2009, Karachi*. IEEE, 2009, ISBN. 978-1-4244-4609-4, pp. 223-228. [Online]. Available:http://ieeexplore.ieee.org/xpl/articleDetails.jsp? arnumber=52671862009.[Accessed: Jun. 12, 2014].

[29] "Quantitative and Qualitative Research," 2010. [Online]. Available: http://www.socsci.uci. edu/ssarc/pcs/webdocs/QuantitativeandQualitativeResearch.pdf.[Accessed: Jun. 17, 2015]

[30] O. Hazzan, "Qualitative Research in Software Engineering," *Department of Education in Technology and Science Technion*, 2010. [Online]. Available: http://edu.technion.ac.il/faculty/orith/homepage/FrontierColumns/OritHazzan_SystemDesigFrontier_Column8.pdf. [Accessed: Jun. 17, 2015].

[31] B. Hancock, "An Introduction to Qualitative Research," 2010. [Online]. Available: http://www.trentrdsu.org.uk/cms/uploads/Qualitative%20Research.pdf, 2010. [Accessed: Jun. 19, 2015]

[32] S. Singh, "Emergence of Payment Systems in the Age of Electronic Commerce: The State of Art," in *Proc. of the 1st Asian Himalayas Int. Conf. on Internet, AH-ICI, 3-5 Nov, 2009, Kathmandu*. IEEE, 2009, ISBN. 978-1-4244-4570-7, pp. 1-18. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5340318. [Accessed: Jun. 19, 2015].

[33] A.A. Albahdal and T.E. Boult, "Problems and Promises of Using the Cloud and Biometrics," in *Proc. of the 11th International Conference on Information Technology: New Generations, ITNG, 7-9 Apr., 2014*, *Las vegas, NV*. IEEE, 2014, ISBN. 978-1-4799-3187-3, pp. 293-300. [Online]. Available: http://ieeexplore.ieee.org/stamp/ stamp.jsp?tp=&arnumber=6822213. [Accessed: Jun. 20, 2015].

[34] O. Ogundele, P. Zavarsky, R. Rahul, and D. Lindskog, "The Implementation of a Full EMV Smartcard for a Point-of-Sale Transaction and its Impact on the PCI DSS," in *Proc. of the Int. Conf. on Social Computing: Privacy, Security, Risk and Trust, PASSAT, 3-5 Sep., 2012, Amsterdam*. IEEE, 2012, ISBN. 978-1-4673-5638-1, pp. 797-806. [Online]. Available: http://ieeexplore.ieee.org/xpl/ articleDetails.jsp?arnumber=6406326. [Accessed: Jun. 23, 2015].

[35] S. Al-Fedaghi, "System-based Approach to Software Vulnerability," in *Proc. of the IEEE Int. Conf. on Social Computing, 20-22 Aug., 2010, Minneapolis, MN*. IEEE, 2010, ISBN. 978-0-7695-4211-9, pp. 1072-1079. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5590497. [Accessed: Jun. 24, 2015].

[36] MAGTEK®, "MAGNETIC STRIPE CARD STANDARDS," [Online]. Available: http://www.magtek.com/documentation/public/99800004-1.08.pdf. [Accessed: Mar. 23, 2016].

[37] J. Liu, Y. Xiao, H. Chen, and S. Ozdemir, "A Survey of Payment Card Industry Data Security Standard," *IEEE Communications Surveys & Tutorials*, *3 Aug., 2010*, vol. 12, no. 3, ISSN. 1553-877X, pp. 287-303. [Online]. Available: http://ieeexplore.ieee.org/xpl/ freeabs_all.jsp?arnumber=5455788. [Accessed: Jun. 25, 2015]

[38] A. H. Qureshi and A.A. Malik, "Evolution of Prepaid Payment Processor's Software Architecture-An Empirical Study," in *Proc. of the 10th International Conference on Frontiers of Information Technology, FIT, 17-19, Dec., 2012, Islamabad*. IEEE, 2012, ISBN. 978-1-4673-4946-8, pp. 188-195. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6424320. [Accessed: Jun. 26, 2015]

[39] A.M. Rashwan, A.M. Taha, and H.S. Hassanein, "Benchmarking Message Authentication Code Functions for Mobile Computing," in *Proc. of the Global Communications Conference, GLOBECOM, 3-7 Dec., 2012, Anaheim, CA*. IEEE, 2012, ISBN. 978-1-4673-0919-6, pp. 2585-2590. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6503506. [Accessed: Mar. 23, 2016]

[40] POS Industry Task Force, 2007. "Best Practices for Point of Sale Lifecycle Security,"[Online]. Available: http://www.ingenicous.com/INGENICO_GALLERY

_CONTENT/Documents/ USA/Best%20Practices%20for%20Point%20of%20Sale %20Security%20-%20Published%20Version%201.pdf.[Accessed: Apr. 23, 2015]

[41] "Skimming Prevention – Best Practices for Merchants," *PCI SSC PIN Transaction Security Working Group,* 2009. [Online]. Available: https://www.pcisecuritystandards.org/ documents/skimming_prevention_IS.pdf. [Accessed: Jun. 26, 2015]

[42] F. Dominici, D. Mazzocchi, P. Mulassano, M. Spelat, G. Boiero, and P. Lovisolo, "NAV/COM Hybrid Architecture for Innovative Location Based Payment Systems," in *Proc. of the Int. Conf. on the 6th IEEE Consumer Communications and Networking Conference, CCNC, 10-13 Jan., 2009 Las Vegas, NV*. IEEE, 2009, ISBN. 978-1-4244-2309-5, pp. 1-5. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4784929. [Accessed: Jul. 2, 2014]

[43] N. Munjal and R. Moona, "Secure and cost effective transaction model for financial services," in *Proc. of the Int. Conf. on Ultra Modern Telecommunications & Workshops, ICUMT,12-14 Oct., 2014, St. Petersburg,* IEEE, 2014, ISBN. 978-1-4244-3941-6, pp. 1-6. [Online]. Available: http://ieeexplore.ieee.org/stamp/ stamp.jsp? tp=&arnumber=5345473. [Accessed: Jul. 2, 2015].

[44] J. Vijayan, 2014. "5 issues that could hamper EMV smartcard adoption in the U.S.," [Online]. Available: http://www.computerworld.com/article/2487581/endpoint-security/5-issues-that-could-hamper-emv-smartcard-adoption-in-the-u-s-.html. [Accessed: May 5, 2015]

[45] "Diebold Introduces New Skimming-Protection Solutions to Combat ATM Fraud," PR Newswire 2015 [Online]. Available: http://www.prnewswire.com/news-releases/ diebold-introduces-new-skimming-protection-solutions-to-combat-atm-fraud-95313419.html.[Accessed: Jul. 2, 2015]

[46] K. Gandhi, 2010. "MagnePrint: A Real Time Risk Management Tool," [Online]. Available: http://www.magneprint.com/docs/99875279-3.01%20MP_WP_print.pdf. [Accessed: Jul. 2, 2015]

[47] H. Lasisi, A.A. Ajisafe, "Development of stripe biometric based fingerprint authentications systems in automated teller machines," in *Proc. of 2012 second Int. Conf. on Advances in Computational Tools for Engineering Applications, ACTEA, 12-15 Dec. 2012 Beirut.* IEEE. [Online]. Available: http://ieeexplore.ieee.org/xpl /articleDetails.jsp?arnumber=6462860.[Accessed: Jul. 2, 2015]

[48] C. Li, M. Hwang, "An efficient biometrics based user authentication scheme using smart cards," *Journal of Network and Computer Applications*, *ScienceDirect,* vol. 11, pp 1–5, Jan. 2010, [Online]. Available: http://www.sciencedirect.com/science/ article/pii/ S1084804509001192. [Accessed: Jul. 2, 2015].

[49] Z. Yongbin, Y. Fucheng, and L Huaqun "Behavior-Based Credit Card Fraud Detecting Model," in *Proc. of the 5th Int. Joint Conf. on INC, IMS and IDC, NCM, 25-27 Aug., 2009, Seoul.* IEEE, 2009, ISBN. 978-0-7695-3769-6, pp. 855-858. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5331646.[Accessed: Jul. 2, 2015]

[50] Y. Makihara, M.A. Hossain, and Y. Yagi, "How to Control Acceptance Threshold for Biometric Signatures with Different Confidence Values?," in *Proc. of the 20th Int. Conf. on Pattern Recognition, ICPR, 23-26 Jul. 4, 2010, Istanbul*. IEEE, 2010, ISBN. 978-1-4244-7542-1, pp. 1208-1211. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5597068. [Accessed: Jul. 3, 2015]

[51] M.U. Munir and M.Y. Javed, "Fingerprint Matching using Ridge Patterns," in *Proc. of the 5th International Conference on Information and Communication Technologies, ICICT, 27-28 Aug., 2005, Pakistan*. IEEE, 2005, ISBN. 0-7803-9421-6, pp. 116-120. [Online]. Available:http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1598565.[Accessed: Jul. 14, 2015]

[52] Y.H.B. Yahaya, M. Isa, and M.I. Aziz "Fingerprint Biometrics Authentication on Smart Card," in *Proc. of the 2nd Int. Conf. on Computer and Electrical Engineering, ICCEE, 28-30 Dec., 2009, Dubai*. IEEE, 2009, ISBN. 978-0-7695-3925-6, pp. 671-673. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5380542.[Accessed: Jul. 14, 2015]

[53] "Fingerprint Identification," 2009. [Online]. Available: http://www.policensw.com/info/fingerprints/finger08.html. [Accessed: May. 22, 2015]

[54] S. Li, "Fingerprint Combination for Privacy Protection," *IEEE Transactions on Information Forensics and Security, 10 Jan., 2013*, vol. 8, no. 2, ISSN. 1556-6013, pp. 350-360. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6384734. [Accessed: Jul. 17, 2015]

[55] J. Thornton, "Setting Standards In The Comparison and Identification," 2000. [Online]. Available: http://www.latent-prints.com/Thornton.htm. [Accessed: Jul. 18, 2015]

[56] M.S. Sadi and T. Kanij, "Fingerprint Verification: A Comparison of Three Approaches," in *Proc. of the Defense Science Research Conference and Expo, DSR, 3-5 Aug., 2011, Singapore*. IEEE, 2011, ISBN. 978-1-4244-9276-3, pp. 1-5. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6026806. [Accessed: Jul. 18, 2015]

[57] J. Tian and Y. Peng, "Research of the Matlab application in the fingerprint identification system," in *Proc. of the Int. Conf. on Image Analysis and Signal Processing, IASP, 9-11 Nov., 2012, Hangzhou*. IEEE, 2012, ISBN. 978-1-4673-2547-9, pp. 1-5. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6425005. [Accessed: Jul. 19, 2015]

[58] A.A. Darwish, W.M. Zaki, O.M. Saad, N.M. Nassar, and G. Schaefer, "Human Authentication using Face and Fingerprint Biometrics," in *Proc. of the 2nd Int. Conf. on Computational Intelligence, Communication Systems and Networks, CICSyN, 28-30 Jul., 2010, Liverpool*. IEEE, 2010, ISBN. 978-0-7695-4158-7, pp. 274-278. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5616134. [Accessed: Jul. 19, 2015]

[59] F. Al-Harby, R. Qahwaji, and M. Kamala, "The effects of gender differences in the acceptance of biometrics authentication systems within online transaction," in *Proc. of the Int. Conf. on CyberWorlds, CW, 7-11 Sep., 2009, Bradford*. IEEE, 2009, ISBN. 978-0-7695-3791-7, pp. 203-210. [Online]. Available: http://ieeexplore.ieee.org/xpl/ articleDetails.jsp?arnumber=5279605. [Accessed: Jul. 20, 2015]

[60] R.A Rajan, N. Sudha, and P.A. Kumar, "O-F estimation based on curved Gabor Filter for fingerprint image enhancement," in *Proc. of the 5th Int. Conf. on Advanced Computing, ICoAC, 18-20 Dec., 2013, Chennai*. IEEE, 2013, ISBN. 978-1-4799-3447-8, pp. 405-412. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6921985. [Accessed: Jul. 21, 2015]

[61] R. Cappelli, M. Ferrara, D. Maltoni, and F. Turroni, 2011. "Fingerprint Verification Competition at IJCB2011," [Online]. Available: http://www.csis.pace.edu/~ctappert/dps/2011IJCB/papers/387.pdf [Accessed: Jul. 21, 2015]

[62] National Institute of Standards and Technology (NIST), 2004. "Fingerprint Vendor Technology Evaluation 2003: Summary of Results and Analysis Report," [Online].Available: http://biometrics.nist.gov/cs_links/fpvte/report/ir_7123_summary.pdf. [Accessed: Jul. 22, 2015]

[63] National Institute of Standards and Technology (NIST), 2003. "Face Recognition Vendor Test 2002: Evaluation Report," [Online]. Available: http://www.face-rec.org/vendors/FRVT_2002_Evaluation_Report.pdf.[Accessed: Jul. 21, 2015]

[64] N.K. Ratha, J.H. Connell, and R.M. Bolle, "An Analysis of Minutiae Matching Strength," in *Proc. of the 3rd Int. Conf. on Audio- and Video-Based Biometric Person Authentication, AVBPA, 6-8 Jun., 2001, Halmstad, Sweden*. Springer Berlin Heidelberg, 2001, ISBN. 978-3-540-42216-7, pp. 223-228. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.87.8743.2001. [Accessed: Jul. 22, 2015]

[65] N.K Ratha, S. Chikkerur, J.H. Connell, and R.M. Bolle, "Generating Cancelable Fingerprint Templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20 Feb., 2007*. IEEE, 2007, vol. 29, no. 4, ISSN. 0162-8828, pp. 561-572.[Online].Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4107561. [Accessed: Jul. 22, 2015]

[66] C. Rathgeb and A. Uhl, "A survey on biometric cryptosystems and cancelable biometrics," 2011. [Online]. Available: http://jis.eurasipjournals.com/content/2011/1/3.[Accessed: Jul. 24, 2015]

[67] P.V. Reddy, A. Kumar, S. Rahman, and T.S. Mundra, "A New Antispoofing Approach for Biometric Devices," *IEEE Transactions on Biomedical Circuits and Systems, 18 Nov., 2008*.IEEE, 2008, vol. 2, no. 4, ISSN. 1932-4545, pp. 328-337, . [Online].Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4669626. [Accessed: Jul. 24, 2015]

[68] A.K Jain, K. Nandakumar, and A. Nagar, "Biometric Template Security," 2008. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?

doi=10.1.1.378.9364&rep=rep1&type=pdf. [Accessed: Jul. 24, 2015]

[69] K. Takahashi and S. Hirata, "Generating Cancelable Fingerprint Templates," in *Proc. of the 3rd Int. Conf. on Biometrics: Theory, Applications, and Systems, BTAS, 28-30 Sep., 2009, Washington, DC*. IEEE, 2009, ISBN. 978-1-4244-5020-6, pp. 1-6. [Online].Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4107561. [Accessed: Jul. 25, 2015]

[70] C. Rathgeb and A. Uhl, "A survey on biometric cryptosystems and cancelable biometrics," 2011. [Online]. Available: http://jis.eurasipjournals.com/content/2011/1/3.[Accessed: Jul. 25, 2015]

[71] V.E. Brindha, "Biometric Template Security using Fuzzy Vault," *IEEE 15th Int. Symposiyum on Consumer Electronics, ISCE, 14-17 Jun., 2011, Singapore*. IEEE, 2011, ISSN. 0747-668X, pp. 384-387. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4669626. [Accessed: Jul. 27, 2015]

[72] E. Marasco and C. Sansone, "An anti-spoofing technique using multiple textural features in fingerprint scanners," in *Proc. of the IEEE Workshop on Biometric Measurements and Systems for Security and Medical Applications, BIOMS, 9-9 Sep., 2010 Taranto*. IEEE, 2010, ISBN. 978-1-4244-6302-2, pp. 8-14 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5610440. [Accessed: Jul. 27, 2015]

[73] A. Antonelli, R. Cappelli, D. Maio, and D. Maltoni, "Fake Finger Detection by Skin Distortion Analysis," *IEEE Transactions on Information Forensics and Security*. IEEE, 2006, vol. 1, no. 3, ISSN. 1556-6013, pp. 360 - 373. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1673397. [Accessed: Jul. 27, 2015]

[74] Y. Satcu, H.T. Sencar, and N. Memon, "A Secure Biometric Authentication Scheme Based on Robust Hashing," 2005. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.91.3167. [Accessed: Jul. 28, 2015]

[75] N. Nishiuchi and H. Soya, "Cancelable Biometric Identification by Combining Biological Data with Artifcats," in *Proc. of the Int. Conf. on Biometrics and Kansei Engineering, ICBAKE, 19-22 Sep., 2011, Takamatsu, Kagawa*. IEEE, 2011, ISBN. 978-0-7695-4512-7, pp. 61-64. [Online]. Available:

http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6031251.
[Accessed: Jul. 28, 2015]

[76] A.M. Canuto, F. Pintro, A.F. Neto, and M.C. Fairhurst, "Enhancing Performance of Cancellable Fingerprint Biometrics using Classifier Ensembles," in *Proc. of the 11th Brazilian Symposium on Neural Networks, SBRN, 23-28 Oct., 2010, Sao Paulo*. IEEE, 2010, ISBN. 978-0-7695-4210-2, pp. 55-60. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5715213. [Accessed: Jul. 29, 2015]

[77] H. Yang, X. Jiang, and A.C. Kot "Generating Secure Cancelable Fingerprint Templates Using Local and Global Features," in *Proc. of the 2nd IEEE Int. Conf. on Computer Science and Information Technology, ICCSIT, 8-11 Aug., 2009, Beijing*. IEEE, 2009, ISBN. 978-1-4244-4520-2, pp. 645-649. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5234870. [Accessed: Jul. 30, 2015]

[78] P.P. Paul and M. Gavrilova, "Multimodal Cancelable Biometrics," in *Proc. of the 11th IEEE Int. Conf. on Cognitive Informatics & Cognitive Computing (ICCI*CC), 22-24 Aug., 2012, Kyoto*. IEEE, 2012, ISBN. 978-1-4673-2794-7, pp. 43-49. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6311208. [Accessed: Jul. 30, 2015]

[79] M. Grassi and M. Faundez-Zanuy, "A protection scheme for enhancing biometric template security and discriminability," 2008. [Online]. Available: http://jrbp10.unizar.es/papers/S1.C1.pdf.[Accessed: Feb 18, 2015]

[80] E. Chandra and K. Kanagalakshmi, "Cancelable Biometric Template Generation and Protection Schemes: a review," in *Proc. of the 3rd Int. Conf. on Electronics Computer Technology, ICECT, 8-10 Apr., 2011, Kanyakumari*. IEEE, 2011, ISBN. 978-1-4244-8679-3, pp. 15-20. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5941948. [Accessed: May 25, 2015]

[81] U. Uludag, S. Pankanti, S. Prabhakar, and A.K. Jain, "Biometric Cryptosystems: Issues and Challenges," in *Proc. of the IEEE, 18 May, 2004*. IEEE, 2004, vol. 92, no. 6, ISSN. 0018-9219, pp. 948-960. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1299169. [Accessed: May 24, 2015]

[82] A. Nagar, K. Nandakumar, and A.K Jain, "Biometric Template Transformation: A Security Analysis," 2008. [Online]. Available: http://www.cse.msu.edu/biometrics/Publications/SecureBiometrics/NagarTempTransSecAnalysis_SPIE10.pdf. [Accessed: Mar. 23, 2016]

[83] C. S. Chin, A. B. J. Teoh, and D. C. L. Ngo, "High Security Iris Verification System Based On Random Secret Integration," Computer Vision and Image Understanding, vol. 102, no. 2, pp.169–177, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S107731420600004X. [Accessed: Mar. 23, 2016]

[84] T. Connie, A. B. J. Teoh, M. Goh, and D. C. L. Ngo, "PalmHashing: A Novel Approach for Cancelable Biometrics," Information Processing Letters, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020019004002741. [Accessed: Mar. 23, 2016]

[85] M. Savvides and B. V. K. Vijaya Kumar, "Cancellable Biometric Filters for Face Recognition," in *Proc. of the IEEE Int. Conference Pattern Recognition*, vol. 3, Cambridge, 2004, UK, pp. 922–925. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1334679. [Accessed: Mar. 23, 2016]

[86] Y. Lee, Y. Chung, and K. Moon, "Inverse Operation and Preimage Attack on BioHashing," in *Proc. of the IEEE Workshop on Computational Intelligence in Biometrics: Theory, Algorithms, and Applications, CIB, Mar. 30, 2009, Nashville, TN*. IEEE, 2009, ISBN. 978-1-4244-2773-4, pp. 92-97. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4925692. [Accessed: May 27, 2015]

[87] N.D. Sarier, "Practical Multi-factor Biometric Remote Authentication," in *Proc. of the Fourth IEEE Int. Conf. on Biometrics: Theory Applications and Systems, BTAS, 27-29 Sep., 2010, Washington, DC*. IEEE, 2010, ISBN. 978-1-4244-7580-3, pp. 1-6. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5634541. [Accessed: May 19, 2015]

[88] N. Radha and S. Karthikeyan, "An Evaluation of fingerprint security using Noninvertible Biohash," 2011. [Online]. Available: http://airccse.org/journal/nsa/0711ijnsa11.pdf. [Accessed: Aug. 16, 2015]

[89] D. Davaanaym, Y.S. Lee, H. Lee, S. Lee, and H. Lim, "A Ping Pong based One-Time Passwords Authentication System," in *Proc. of the 5th Int. Joint Conf. on INC, IMS and IDC, NCM, 25-27 Aug., 2009, Seoul*. IEEE, 2009, ISBN. 978-0-7695-3769-6, pp. 574-579. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5331787. [Accessed: Aug. 2, 2015]

[90] A.A. Jilani, A. Nadeem, T.H. Kim and E.S. Cho, "Formal Representations of the Data Flow Diagram: A Survey," in *Proc. of the Int. Conf. on Advanced Software Engineering and Its Applications, ASEA*, *13-15 Dec., 2008, Hainan Island*. IEEE, 2008, ISBN. 978-0-7695-3432-9, pp. 153-158. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4721332. [Accessed: Aug. 3, 2015]

[91] Payment Card Industry (PCI) Data Security Standard, "Requirements and Security Assessment Procedures Version 2.0," 2010. [Online]. Available: https://www.pcisecuritystandards.org/documents/pci_dss_v2.pdf. [Accessed: Aug. 10, 2015]

[92] American National Standards for Financial Services- ANS X-9.24 -2002, 2010. "Retail Financial Services Symmetric Key Management Part1: Using Symmetric Techniques," [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso:11568:-2:ed-3:v1:en.[Accessed: Aug. 10 12, 2013]

[93] J. Malik, D. Girdhar, R. Dahiya, and G. Sainarayanan, "Reference Threshold Calculation for Biometric Authentication," 2014. [Online]. Available: http://www.mecs-press.org/ijigsp/ijigsp-v6-n2/IJIGSP-V6-N2-6.pdf. [Accessed: Mar. 23, 2016]

[94] T. Nipkow, L.C. Paulson, and M. Wenzel, "Isabelle/HOL A Proof Assistant for Higher-Order Logic," 2014. [Online]. Available: http://isabelle.in.tum.de/doc/tutorial.pdf.[Accessed: Aug. 11, 2015]

[95] G. Lowe, P. Broadfoot, C. Dilloway, and M.L. Hui, "Casper A compiler for the analysis of security protocols," 2009.[Online]. Available: http://www.cs.ox.ac.uk/gavin.lowe/Security/Casper/manual.pdf. [Accessed: Aug. 11, 2015]

[96] M. Hussein and D. Seret, "Extending TLS for trust delegation in home networks," 2006. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.108.6958&rep=rep1&type=pdf. [Accessed: Aug. 11, 2015]

[97] J. Wei, G. Su, and M. Xu, "An Integrated Model to Analyze Cryptographic Protocols with Colored Petri Nets," in *Proc. of the 11th IEEE. High Assurance Systems Engineering Symposium, HASE, 3-5 Dec., 2008 Nanjing*. IEEE, 2008, ISBN. 978-0-7695-3482-4, pp. 457-460. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4708906. [Accessed: May 13, 2015]

[98] R. Cooke and R. Anane, "Robust e-Voting Composition," in *Proc. of the 17th IEEE Int. Conf. on Parallel and Distributed Systems, ICPADS, 7-9 Dec., 2011 Tainan*. IEEE, 2011, ISBN. 978-1-4577-1875-5, pp. 676-683. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6121340. [Accessed: Aug. 14, 2015]

[99] C. Cremers and P. Lafourcase, 2009. "Comparing State Spaces in Automatic Security Protocol Verification," [Online]. Available: http://www.lsv.ens-cachan.fr/Publis/PAPERS/PDF/CL-avocs07.pdf. [Accessed: Aug. 25, 2015]

[100] B. Blanchet, B. Smith, and V. Cheval, "ProVerif 1.90: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial," 2015. [Online]. Available: http://prosecco.gforge. inria.fr/personal/bblanche/proverif/manual.pdf. [Accessed: Sep. 22, 2015]

[101] M. Nesi and G. Rucci, "Formalizing and Analysing the Needham-Schroeder Symmetric-Key Protocol by Rewriting," 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1571066105050541#. [Accessed: Sep. 22, 2015]

[102] S. Shaikh and V. Bush, "Analysing the Woo-Lam Protocol Using CSP and Rank Functions," Journal of Research and Practice in Information Technology, 2006. [Online].Available: https://www.acs.org.au/__data/assets/pdf_file/0008/15389/JRPIT38.1.19.pdf [Accessed: Sep. 22, 2015]

[103] I. Cervasato, C. Meadows, and D. Pavlovic, "Deriving Key Distribution Protocols and their Security Properties," 2006. [Online]. Available: http://reports-archive.adm.cs.cmu.edu/anon/2006/CMU-CS-06-172.pdf [Accessed: Sep. 22, 2015]

[104] L. Chen and M. Shi, "Security analysis and improvement of Yahalom protocol," in *Proc. of the 3rd IEEE Conference on Industrial Electronics and Applications, ICIEA, 3-5 Jun., 2008 Singapore*. IEEE, 2008, ISBN. 978-1-4244-1718-6, pp.

1137-1140. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4582696. [Accessed: Sep. 22, 2015]

[105]L. Lei-jun., "Improved Yahalom protocol analysis theory based on ideal and honest," in *Proc. of the Int. Conf. on Computer Design and Applications, ICCDA, 25-27 Jun., 2010 Qinhuangdao*. IEEE, 2010, ISBN. 978-1-4244-7164-5, pp. V4-317-V4-320. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5541122. [Accessed: Sep. 12, 2015]

[106] B. Blanchet and M. Chaudhuri, "Automated Formal Analysis of a Protocol for Secure File Sharing on Untrusted Storage," in *Proc. of the IEEE Symposium on Security and Privacy, SP, 18-22 May, 2008, Oakland*, *CA*. IEEE, 2008, ISBN. 978-0-7695-3168-7, pp. 417-431. [online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4531168. [Accessed: Sep. 11, 2015]

[107] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Plutus: Scalable Secure File Sharing on Untrusted Storage," in *Proc. of the 2nd USENIX Conf. on File and Storage Technologies, FAST, Berkley.* ACM DL, 2003, pp. 29-42. [Online]. Available: http://dl.acm.org/citation.cfm?id=1090698. [Accessed: Sep. 12, 2015]

[108] M. Abadi and B. Blanchet, "Computer-Assisted Verification of a Protocol for Certified Email," 2005. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.15.325. [Accessed: Sep. 13, 2015]

[109] K. Bhargavan, C. Fournet, A.D. Gordon, and S. Tse "Verified Interoperable Implementations of Security Protocols," in *Proc. of the 19th IEEE Computer Security Foundations Workshop, 2006*, *Venice.* IEEE, 2006, ISSN. 1063-6900, pp. 14-152. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1648714. [Accessed: Sep. 14, 2015]

[110] D. Raluca and B. Monica., "TLS Protocol: Improvement Using Proxies," in *Proc. of the 10th Int. Symposium on Electronics and Telecommunications, ISETC, 15-16 Nov., 2003, Timisoara.* IEEE, 2003, ISBN. 1978-1-4673-1177-9, pp. 151-154

[online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6408094.[Accessed: Sep. 15, 2015]

[111] K. Bhargavan, C. Fournet, A.D. Gordon, and N. Swamy, "Verified Implementations of the Information Card Federated Identity-Management Protocol," 2008. [Online]. Available: http://www.cs.umd.edu/~nswamy/papers/wcf-cardspace.pdf. [Accessed: Sep. 16, 2015]

[112] K. Bhargavan and R. Corin, "Cryptographically Verified Implementations of TLS," 2008.[Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.153.3950&rep=rep1&type=pdf. [Accessed: Sep. 17, 2015]

[113] M. Abadi, B. Blanchet, and C. Fournet, "Just Fast Keying in the Pi Calculus," 2004. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.137.2471. [Accessed: Sep. 17, 2015]

[114] W. Aiello, S.M. Bellovin, M. Blaze, R. Canetti, J. Loannidis, and A.D. Keromytis, "Just Fast Keying: Key Agreement in a Hostile Internet," 2009. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.423.463. [Accessed: Sep. 18, 2015]

[115] S. Delaune, S. Kremer, and M. Ryan, "Verifying Privacy-Type Properties of Electronic Voting Protocols", 2009. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.142.1731. [Accessed: Sep. 20, 2015]

[116] S. Kremer and M.D. Ryan, "Analysis of Electronic Voting Protocol in the Applied Pi Calculus," 2008. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.59.325. [Accessed: Sep. 21, 2015]

[117] M. Backes, C. Hritcu, and M. Maffei, "Automated Verification of Remote Electronic Voting Protocols in the Applied Pi-Calculus," in *Proc. of the 21st IEEE Computer Security Foundations Symposium, CSF, 23-25 Jun., 2008, Pittsburgh, PA, USA*. IEEE, 2008, ISBN. 978-0-7695-3182-3, pp. 195-209. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4556687. [Accessed: Sep. 22, 2015]

[118] S. Delaune, M. Ryan, and B. Smyth, "Automatic Verification of Privacy Properties in the applied pi calculus," 2008. [Online]. Available:

http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.131.6396.
[Accessed: Sep. 23, 2015]

[119] B. Smyth, M.Ryan, and L. Chen, "Direct Anonymous Attestation (DAA): Ensuring privacy with corrupt administrators," in *Proc. of the 4th European conference on Security and privacy in ad-hoc and sensor networks, ESAS, 2007, Berlin.* ACM DL, 2007, ISBN. 978-3-540-73274-7, pp. 218-231. [Online]. Available: http://dl.acm.org/citation.cfm?id=1784426.[Accessed: Sep. 24, 2015]

[120] M. Backes, M. Maffei, and D. Unruh, "Zero-Knowledge in the Applied Pi-calculus and Automated Verification of the Direct Anonymous Attestation Protocol," in *Proc. of the IEEE Symposium on Security and Privacy, SP, 18-22 May, 2008 Oakland, CA*. IEEE, 2008, ISBN. 978-0-7695-3168-7, pp. 202-215. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4531154.
[Accessed: Sep. 24, 2015]

[121] R. Kusters and T. Truderung, "Using ProVerif to Analyze Protocols with Diffie-Hellman Exponentiation," in *Proc. of the 22nd IEEE Computer Security Foundations Symposium, CSF, 8-10 Jul., 2009, Port Jefferson, NY*. IEEE, 2008, ISBN. 978-0-7695-3712-2, pp. 151-171. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5230620.
[Accessed: Sep. 25, 2015]

[122] L. Chen and M. Ryan, "Attack, Solution and Verification for Shared Authorisation Data in TCG TPM," in *Proc. of the 6th international conference on Formal Aspects in Security and Trust, FAST, 2009, Berlin.* ACM DL, 2009, ISBN. 3-642-12458-5 978-3-642-12458-7, pp. 201-216. [Online]. Available: http://dl.acm.org/citation.cfm?id=2128256. [Accessed: Oct. 1, 2015]

[123] B. Smyth, M. Ryan, S. Kremer, and M. Kourjieh, "Towards automatic analysis of election verifiability properties," 2009. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.153.5726.
[Accessed: Oct. 1, 2015]

[124] S. Kremer, M. Ryan, and B. Smyth, "Election verifiability in electronic voting protocols," in *Proc. of the 15th European conference on Research in computer security, ESORICS, 2010, Berlin.* ACM DL, ISBN. 3-642-15496-4 978-3-642-15496-6, pp. 389-404. [Online]. Available: http://dl.acm.org/citation.cfm?id=1888912.[Accessed: Oct. 2, 2015]

[125] M. Arapinis, T. Chothia, E. Ritter, and M. Ryan, "Analysing Unlinkability and Anonymity Using the Applied Pi Calculus," in *Proc. of the 23rd IEEE Computer Security Foundations Symposium, 17-19 Jul., 2010, Pittsburgh*. IEEE, 2010, ISBN. 978-1-4244-7511-7, pp. 107-121. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5552649. [Accessed: Oct. 2, 2015]

[126] J.M. Wing, " FAQ on π-Calculus," 2002. [Online]. Available: http://www.cs.cmu.edu/~wing/publications/Wing02a.pdf.[Accessed: Oct. 3, 2015]

[127] A. Salaiwarakul and M.D. Ryan, "Verification of Integrity and Secrecy Properties of a Biometric Authentication Protocol," 2008. [Online]. Available: http://www.cs.bham.ac.uk/~mdr/research/papers/pdf/08-biometric.pdf. [Accessed: Sep. 6, 2015]

[128] B.S. Kurhade and M. Kshirsagar, "Formalization and Analysis of Borda protocol using pi calculus," in *Proc. of the Int. Conf. on Pattern Recognition, Informatics and Mobile Engineering, PRIME, 21-22 Feb., 2013, Salem*. IEEE, 2013, ISBN. 978-1-4673-5843-9, pp. 232-236. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6496478. [Accessed: Sep. 17, 2015]

[129] S. Delaune, S. Kremer, and M. Ryan "Coercion-resistance and Receipt freeness in Electronic Voting," in *Proc. of the 19th IEEE Computer Security Foundations Workshop, 2006, Venice*. IEEE, 2006, ISBN. 0-7695-2615-2, pp. 12-42. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1648706. [Accessed: Sep. 18, 2015]

[130] D. Dolev and A.C. Yao, "On the Security of Public Key Protocols," *IEEE Transactions on Information Theory, 2003*. IEEE, 2003, vol. 29, no. 2, ISSN. 0018-9448, pp. 198-208. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1056650. [Accessed: Jun. 12, 2015]

[131] B. Blanchet, "Automatic Proof of Strong Secrecy for Security Protocols," 2004. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download? doi=10.1.1.127.9080&rep=rep1&type=pdf. [Accessed: Sep. 21, 2015]

[132] F. Liu and M. Yang, "Verification and Validation of AI simulation Systems," in *Proc. of the Int. Conf. on Machine Learning and Cybernetics*, *26-29 Aug., 2004*. IEEE, 2004, vol. 5, ISBN. 0-7803-8403-2, pp. 3100-3105. [Online]. Available:

http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1378566.
[Accessed: Sep. 1, 2015]

[133] International Committee for Information Technology Standards INCITS , "Study Report on Biometrics in E-Authentication," 2007. [Online]. Available: https://standards.incits.org/apps/group_public/download.php/24528/ m1070185rev.pdf. [Accessed Sep. 2, 2015]

[134] "Information Technology– Open Systems Interconnection- Security Frameworks for Open Systems; Authentication Framework", *ITU-T Recommendations ITU-T X.811(04/95),* 2008. [Online]. Available: http://www.itu.int/rec/T-REC-X.811/en.[Accessed: Sep. 3, 2015].

[135] C. Roberts, "Biometric Attack Vectors and Defenses," *Journal of Computers & Security*, ScienceDirect, 2007. vol. 26, no. 1, doi. 10.1016/j.cose.2006.12.008, pp 14–25. [Online]. Available: http://www.sciencedirect.com/science/article/pii/ S016740480600215X.[Accessed: Sep. 4, 2015].

[136] X. Li, L. Hendren, "Mc2 FOR Demo: A Tool for Automatically Translating MATLAB to FORTRAN 95," in *Proc. of the Software Evolution Week - IEEE Conference on Software Maintenance, Reengineering and Reverse Engineering, CSMR-WCRE, 3-6 Feb., 2014, Antwerp.* IEEE, 2014, doi. 10.1109/CSMR-WCRE.2014.6747218, pp. 458-463. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6747218. [Accessed: Sep. 20, 2015].

[137] V.K. Alilou, "Fingerprint Matching: A Simple approach," MATLAB CENTRAL, 2013. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/ 44369-fingerprint-matching-a-simple-approach.[Accessed: Oct. 3, 2015].

[138] H. Yoshimura, "Fingerprint Templates with High Recognition Accuracy and High Security Generated by Discrete Fractional Sine Transform," in *Proc. of the 11th Int. Conf. on Internet Technology and Secured Transactions, ICITST, 11-14 Dec., 2011, Abu Dhabi.* IEEE, 2014, ISBN. 978-1-4577-0884-8, pp. 185-190. [Online]. Available: http://ieeexplore. ieee.org/stamp/stamp.jsp?tp=&arnumber=6148447. [Accessed: Oct. 4, 2015].

[139] G. Vitello, V. Conti, A. Gentile, S. Vitabile, and F. Sorbello, "Design and Implementation of an Efficient Fingerprint Features Extractor," in *Proc. of the 17th Euro Micro Conference on Digital System Design, DSD, 27-29 Aug., 2014, Verona.*

IEEE, 2014, doi. 10.1109/DSD.2014.101, pp. 695-699. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6927316. [Accessed: Oct. 5, 2015].

[140] J. Wayman, A. Jain, D. Maltoni, and D. Maio, "Biometric Systems: Technology, Design and Performance Evaluation,". SPRINGER, 2004, ISBN. 1852335963. [Online]. Available: http://www.amazon.com/Biometric-Systems-Technology-Performance-Evaluation/dp/1852335963.[Accessed: Oct. 6, 2015].

[141] P. Lacharme, "Revising the accuracy of the biohashing algorithm on fingerprints," *Biometrics, IET , 2013*. IEEE, 2013, vol. 11, no. 3, ISSN. 2047-4938, pp. 130-133. [Online]. Available: http://dx.doi.org/10.1049/iet-bmt.2012.0041. [Accessed: Oct. 6, 2015].

[142] D. Kumar and Y. Ryu, "A Brief Introduction of Biometrics and Fingerprint Payment Technology," in *Proc. of the 2nd Int. Conf. on Future Generation Communication and Networking Symposia, FGCNS, 13-15 Dec., 2008, Sanya*. IEEE, 2008, ISBN. 978-0-7695-3546-3, pp. 185-192. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4813576. [Accessed: Oct. 7, 2015].

Appendix

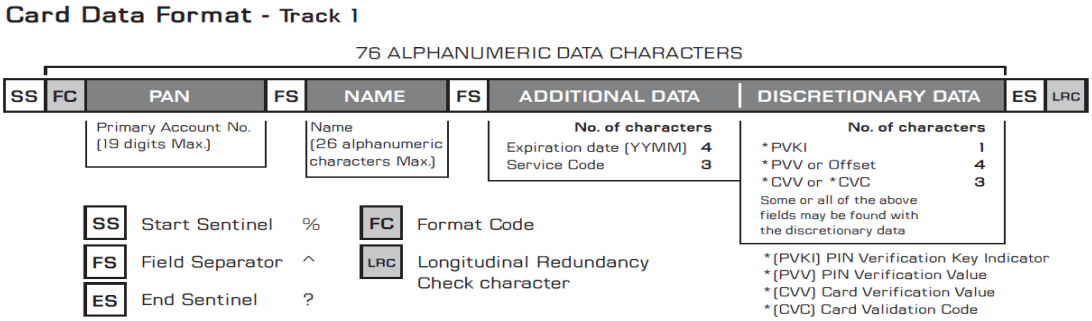


Figure 2.2: Track 1 [36]

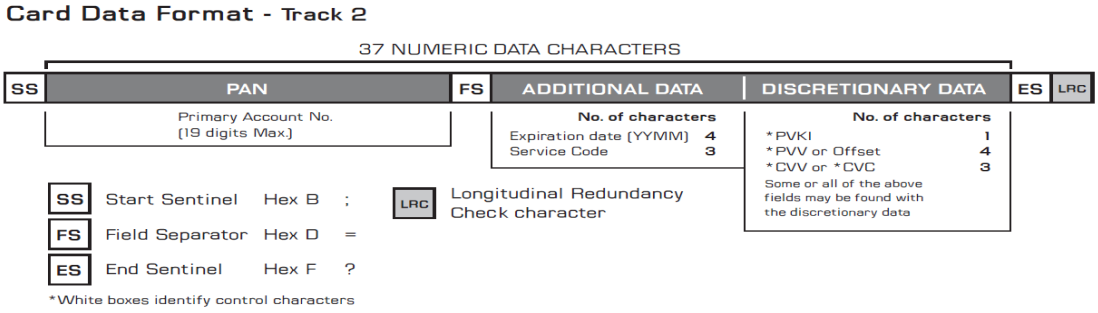


Figure 2.3: Track 2 [36]

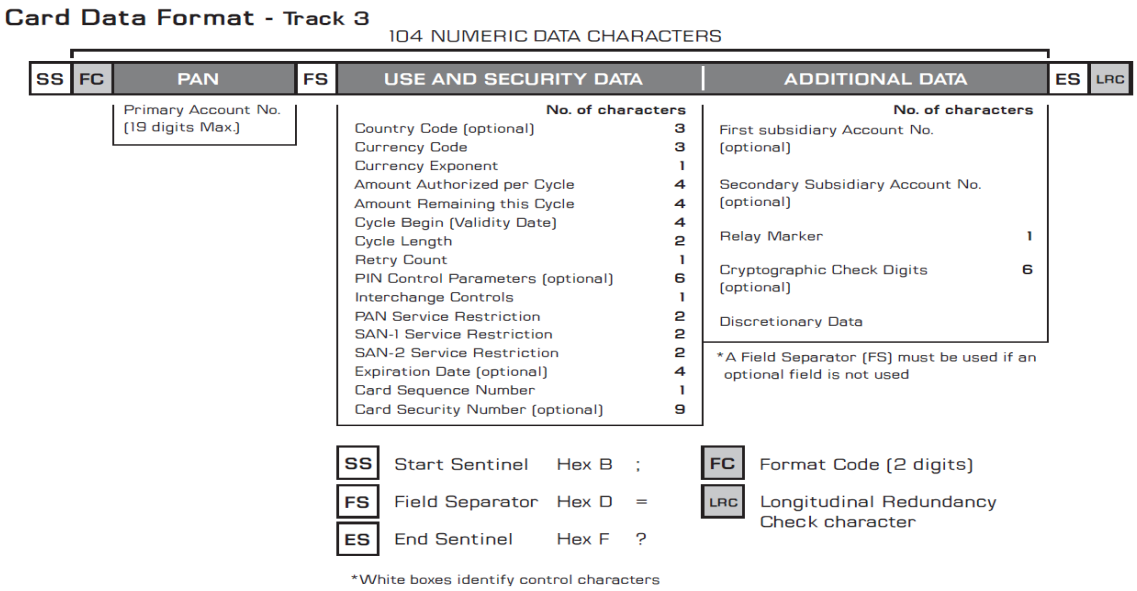


Figure 2.4: Track 3 [36]

Table 6.4: ProVerif script for the implementation of U Process

```
(* UProcess.pv: Script that models the User Process *)

(* to get detailed traces *)
set traceDisplay = long.

(* c represents a public channel for communication *)
free c : channel.

(* bc represents a bounded or restricted channel for communication *)
free bc : channel [private].

(* MSCD represents user's magnetic stripe bank card data *)
free MSCD : bitstring [private].

(* FPReq represents request to present fingerprint template *)
free FPReq : bitstring [private].

(* FPPresented represents the fingerprint template presented *)
free FPPresented : bitstring [private].

(* represents the user's Authentication Status Update success*)
free ASUSuccess: bitstring [private].

(* represents the user's Authentication Status Update failure*)
free ASUFailure: bitstring [private].

(* event declaration *)
event evPresentMSC.
event evCardSwipeFailure.
event evPresentFP.
event evAuthenticationResponseSuccess.
event evAuthenticationFailure.

(* query to test adversary can derive MSCD *)
query attacker (MSCD).

(* query to test adversary can derive fingerprint template*)
query attacker (FPPresented).

(* query to test evAuthenticationResponseSuccess event can occur before
  evPresentFP and evPresentMSC event  *)
query event (evAuthenticationResponseSuccess) ==> event (evPresentFP) &&
                                                   event (evPresentMSC).

(* query to test evAuthenticationFailure event can occur before
  evPresentFP and evPresentMSC event  *)
query event (evAuthenticationFailure) ==> event (evPresentFP) &&
                                          event (evPresentMSC).

(* start of the process*)
process

(* sending MSCD through bounded channel *)
out( bc,MSCD) ;

(* receiving data through bounded channel *)
in ( bc,x:bitstring) ;

(* receiving fingerprint  template present request through bounded channel *)
if x = FPReq then
event evPresentFP;
event evPresentMSC;
(* sending presented fingerprint template bounded channel to POS Process*)
out ( bc,FPPresented)
else
event evCardSwipeFailure;
event evPresentMSC;

(* receiving data through bounded channel *)
in ( bc,y:bitstring) ;

(* receiving AuthenticationStatusUpdate data through bounded channel *)
if y = ASUSuccess then
```

```
74 event evAuthenticationResponseSuccess;
75 event evPresentFP;
76 event evPresentMSC
77 else
78 event evAuthenticationFailure;
79 event evPresentFP;
80 event evPresentMSC
```

Table 6.5: ProVerif script for the implementation of POS Process

```
1 (* POSProcess.pv: Script that models the POS Process *)
2
3  (* to get detailed traces *)
4  set traceDisplay = long.
5
6 (* c represents a public channel for communication *)
7 free c : channel.
8
9 (* bc represents a bounded or restricted channel for communication *)
10 free bc : channel [private].
11
12 (* MSCD represents user's magnetic stripe bank card data*)
13 free MSCD : bitstring [private].
14
15 (* LReq represents the LogonRequest *)
16 free LReq: bitstring [private].
17
18 (* LRespSuccess represents a successful LogonResponse *)
19 free LRespSuccess : bitstring [private].
20
21 (* FPReq represents request to present fingerprint template *)
22 free FPReq : bitstring [private].
23
24 (* FPPresented represents the fingerprint template presented *)
25 free FPPresented : bitstring [private].
26
27 (* AReq represents the AuthenticationRequest *)
28 free AReq : bitstring [private].
29
30 (* ARespSuccess represents the successful AuthenticationResponse *)
31 free ARespSuccess : bitstring [private].
32
33 (* event declarations *)
34 event evMSCPresented.
35 event evCardSwiped.
36 event evCardSwipeFailure.
37 event evLogonResponseSuccess.
38 event evLogonResponseFailure.
39 event evFPPresented.
40 event evAuthenticationResponseSuccess.
41 event evAuthenticationFailure.
42 event evTransactionDeclined.
43 event evTransactionContinue.
44
45 (* query to test adversary can derive MSCD *)
46 query attacker (MSCD).
47
48 (* query to test adversary can derive user's finger template *)
49 query attacker (FPPresented).
50
51 (* query to test evLogonResponseSuccess event can occur before evCardSwiped
52     and evMSCPresented event  *)
53 query event (evLogonResponseSuccess) ==> event (evCardSwiped) &&
54                                                 event (evMSCPresented).
55
56 (* query to test evTransactionDeclined event can occur before
57 evLogonResponseFailure,evCardSwiped and evMSCPresented event  *)
58 query event (evTransactionDeclined) ==> event (evLogonResponseFailure) &&
59  event (evCardSwiped) && event (evMSCPresented).
60
61 (* query to test if evTransactionContinue event can occur before the
62 evAuthenticationResponseSuccess, evLogonResponseSuccess,
63 evCardSwiped and evMSCPresented event *)
64 query event (evTransactionContinue) ==> event (evAuthenticationResponseSuccess) 65  && event (evLogonResponseSuccess)
```

```
&& event (evFPPresented)   &&
66  event (evCardSwiped)   && event (evMSCPresented).
67
68 (* query to test if evTransactionDeclined  event can occur before
69 evAuthenticationFailure, evLogonResponseSuccess, evFPPresented,evCardSwiped 70 and evMSCPresented event  *)
71 query event (evTransactionDeclined) ==> event (evAuthenticationFailure) &&
72  event (evLogonResponseSuccess) &&event (evFPPresented)   &&
73  event (evCardSwiped)   && event (evMSCPresented).
74
75 (* start of the process*)
76 process
77
78 (* receiving MSCD through incoming bounded channel *)
79 in( bc,w:bitstring) ;
80
81 if w = MSCD then
82 (* sending logon request through bounded channel to BAS Process*)
83 out ( bc,LReq)
84 else
85 (* inform U Process about the card swipe failure*)
86 event evCardSwipeFailure;
87
88 (* receiving logon response through bounded channel from BAS Process*)
89 in ( bc,x:bitstring) ;
90
91 if x = LRespSuccess then
92 event evLogonResponseSuccess;
93 event evCardSwiped;
94 event evMSCPresented;
95   (* sending present FP request through bounded channel to U Process  *)
96   out ( bc,FPReq)
97 else
98    event evTransactionDeclined;
99    event evLogonResponseFailure;
100  event evCardSwiped;
101  event evMSCPresented;
102
103 (* receiving FP response through bounded channel from U Process  *)
104  in ( bc,y:bitstring) ;
105
106 if y = FPPresented then
107 (*sending authentication request through bounded channel to BAS Process*)
108 out( bc,AReq)
109 else
110 (* sending evTransactionDeclined to U Process *)
111  event evTransactionDeclined;
112
113 (* receiving authentication response through bounded channel from BAS Process *)
114 in( bc,z:bitstring) ;
115
116 (* receiving AuthenticationResponse data through bounded channel *)
117 if z = ARespSuccess then
118 event evTransactionContinue;
119 event evAuthenticationResponseSuccess;
120 event evLogonResponseSuccess;
121 event evFPPresented;
122 event evCardSwiped;
123 event evMSCPresented
124 else
125 event evTransactionDeclined;
126 event evAuthenticationFailure;
127 event evLogonResponseSuccess;
128 event evFPPresented;
129 event evCardSwiped;
130 event evMSCPresented
```

Table 6.6: ProVerif script for the implementation of BAS Process

```
1  (* BASProcess.pv: Script that models the BAS Process *)
2
3  (* to get detailed traces *)
4  set traceDisplay = long.
5
6  (* c represents a public channel for communication *)
```

```
7  free c : channel.
8
9  (* bc represents a bounded or restricted channel for communication *)
10 free bc : channel [private].
11
12 (* VLReq represents a Valid LogonRequest *)
13 free VLReq: bitstring [private].
14
15 (* LRespSuccess represents a successful LogonResponse *)
16 free LRespSuccess : bitstring [private].
17
18 (* LRespFailure represents an unsuccessful LogonResponse *)
19 free LRespFailure : bitstring [private].
20
21 (* VAReq represents the Valid AuthenticationRequest *)
22 free VAReq : bitstring [private].
23
24 (* ARespSuccess represents the successful AuthenticationResponse *)
25 free ARespSuccess : bitstring [private].
26
27 (* ARespFailure represents an unsuccessful AuthenticationResponse *)
28 free ARespFailure : bitstring [private].
29
30 (* event declarations *)
31 event evLogonRequest.
32 event evLogonResponseSuccess.
33 event evLogonResponseFailure.
34 event evAuthenticationRequest.
35 event evAuthenticationResponseSuccess.
36 event evAuthenticationFailure.
37
38 (* query to test evLogonResponseSuccess event can occur before
39    evLogonRequest event  *)
40  query event (evLogonResponseSuccess) ==> event (evLogonRequest).
41
42 (* query to test evLogonResponseFailure event can occur before
43     evLogonRequest event  *)
44 query event (evLogonResponseFailure) ==> event (evLogonRequest).
45
46 (* query to test if evAuthenticationResponseSuccess event can occur before the
47   evLogonRequest, evLogonResponseSuccess and  evAuthenticationRequest event *)
48 query event (evAuthenticationResponseSuccess) ==> event (evLogonRequest) &&
49 event (evLogonResponseSuccess) && event (evAuthenticationRequest).
50
51 (* query to test if evAuthenticationResponseFailure event can occur before the
52 evLogonRequest, evLogonResponseSuccess and  evAuthenticationRequest event *)
53 query event (evAuthenticationFailure) ==> event (evLogonRequest) &&
54 event (evLogonResponseSuccess) && event (evAuthenticationRequest).
55 (* start of the process*)
56 process
57
58 (* receiving logon request through incoming bounded channel *)
59  in( bc,x:bitstring) ;
60
61 if x = VLReq then
62 event evLogonRequest;
63 (* sending LRespSuccess through bounded channel to POS Process  *)
64  out ( bc,LRespSuccess)
65 else
66 event evLogonResponseFailure;
67 (* sending LRespFailure through bounded channel to POS Process  *)
68 out( bc,LRespFailure);
69
70 (* receiving a valid authentication request through bounded channel from
71 POS  Process  *)
72 in ( bc,y:bitstring) ;
73
74 if y = VAReq then
75 event evAuthenticationResponseSuccess;
76 (* sending ARespSuccess through bounded channel to POS Process  *)
77 out( bc,ARespSuccess)
78 else
79 event evAuthenticationFailure;
80 (* sending ARespFailure through bounded channel to POS Process  *)
81 out( bc,ARespFailure)
```

Table 7.1: Matlab script for the implementation of OTT Algorithm at BAS

```
%Implementation of the OTT_BAS_Algorithm
function [OTT_GENERATED_IN_BAS, ECC] = OTT_BAS(enrolled_template);
clc
disp('Start generation process @ BAS (Biometric Authentication Server) started...');
OTT_GENERATED_IN_BAS = enrolled_template;
%map_attributes_to_weights step
attributes= cell(8,1);
attributes(1,1)= {'7704166087189'};
attributes(2,1)= {'kishor'};
attributes(3,1)= {'40202127'};
attributes(4,1)= {'santhakumari'};
attributes(5,1)= {'55145789'};
attributes(6,1)= {'manju'};
attributes(7,1)= {'keerthana'};
attributes(8,1)= {'kritha'};
disp('The captured attributes for the user follows')
disp(attributes)
disp('Performing map_attributes_to_weights')
max_attributes_len = 16;
for i=1:8
   padded_len = max_attributes_len  - length(attributes{i});
   attributes(i,1) = map_attributes_to_weights(attributes(i),padded_len);
end
disp(attributes)
str = strcat(attributes);
disp(str)
weight1 = dec2bin(str{1},1);weight2 = dec2bin(str{2},1);weight3 = dec2bin(str{3},1);weight4 = dec2bin(str{4},1);
weight5 = dec2bin(str{5},1);weight6 = dec2bin(str{6},1);weight7 = dec2bin(str{7},1);weight8 = dec2bin(str{8},1);
disp('map_attributes_to_weights completed...')
disp('weights are as follows:')
fprintf('weight1:%s\r\n',weight1)
fprintf('weight2:%s\r\n',weight2)
fprintf('weight3:%s\r\n',weight3)
fprintf('weight4:%s\r\n',weight4)
fprintf('weight5:%s\r\n',weight5)
fprintf('weight6:%s\r\n',weight6)
fprintf('weight7:%s\r\n',weight7)
fprintf('weight8:%s\r\n',weight8)
weights = {weight1,weight2,weight3,weight4,weight5,weight6,weight7,weight8};
disp(weights)
%derive Biometric Keys (BKs) step
BK_Temp = [];
for i=1:1000
   BK_Temp = [BK_Temp;sprintf('%s',weights{randperm(length(weights))})];
end
%map the 128 bits of BK_Temp to BK
BK= [];
for i=1:length(BK_Temp)
  BK =[BK;sprintf('%s', BK_Temp(i,1:128))];
end
disp('Biometric Keys of length 128 bits are dervied successfully!!!')
disp('BKs are as follows')
for i=1:length(BK)
   disp(BK(i,:))
end
 %Compute Biometric Template Transformation (BTT) of length 128 bytes
BK_Vault1= [];
BK_Vault2= [];
BK_Final = [];
for i=1:length(BK_Temp)
  BK_Vault1=[BK_Vault1;sprintf('%s', BK(i,1:16))];
  BK_Vault2=[BK_Vault2;sprintf('%s', BK(i,113:128))];
  BK_Final=[BK_Final;sprintf('%s', strcat(BK_Vault1(i,1:16),BK_Vault2(i,1:16)))];
end
disp('displaying sizeof BK_Final')
disp(size(BK_Final))
disp(length(BK_Final))

BK_Finally = [];
for i=1:1000
   BK_Finally=[BK_Finally;sprintf('%s', num2str(BK_Final(i,:)))];
```

```
   disp(BK_Final(i,:))
end
BK_Final_Int = [];
for i=1:1000
   BK_Final_Int=[BK_Final_Int;sprintf('%04i', bin2dec(BK_Finally(i,1:4)))];
   disp(BK_Final_Int(i,:))
end
 [x y] = size(enrolled_template)
enrolled_template_x = [];
enrolled_template_y = [];
for i=1:x
   enrolled_template_x=[enrolled_template_x;sprintf('%04i', enrolled_template(i,1))];
   enrolled_template_y=[enrolled_template_y;sprintf('%04i', enrolled_template(i,2))];
end
TT = [];
disp('Minutiaes follows....')
for i=1:x
   disp([enrolled_template_x(i,:) ',' enrolled_template_y(i,:)])
end
j = 1;
for i=1:1000
   if(mod(i, x) == 0)
       j = 1;
   end              TT=[TT;sprintf('%012s',strcat(BK_Final_Int(i,:),enrolled_template_x(j,:),enrolled_template_y(j,:)))];
   j = j + 1;
end
disp('TT follows....')

for i=1:length(BK_Final)
   disp(TT(i,:))
end
%Generate IOTT by salting TTs and TSNs (Transaction Numbers)
disp('IOTT follows....')
IOTT = [];
for TSN=1:x
   IOTT=[IOTT;sprintf('%016s', strcat(TT(TSN,:),num2str(TSN)))];
   disp(IOTT(TSN,:))
end
%Generate OTT by salting IOTT with DS (DateStamp) and TS (Time Stamp)
disp('OTT follows....')

OTT1 = [];
OTT = [];
TEMP_OTT_GENERATED_IN_BAS = [];

S= datestr(now);
for DS_TS=1:x
   OTT1=[OTT1;sprintf('%032s', strcat(num2str(IOTT(DS_TS,:)),num2str(S)))];%Salting OTT with DS &TS
   %disp(OTT1(DS_TS,:));
   OTT=[OTT;sprintf('%i', OTT1(DS_TS,:))];
   %disp(OTT(DS_TS,:));
   TEMP_OTT_GENERATED_IN_BAS=[TEMP_OTT_GENERATED_IN_BAS;sprintf('%X', OTT(DS_TS,17:19))];
   %disp(TEMP_OTT_GENERATED_IN_BAS(DS_TS,:));
end
temp_template = [];
temp_template = TEMP_OTT_GENERATED_IN_BAS;
input_template = [];
input_template = OTT_GENERATED_IN_BAS;
OTT_GENERATED_IN_BAS = OTT_GENERATION(input_template, temp_template);
 end
```

Table 7.2: Matlab script for the implementation of OTT Algorithm at POS

```
%Implementation of the OTT_POS_Algorithm
function [OTT_GENERATED_IN_POS] = OTT_POS(query_template);
clc
disp('Start generation process @ POS (Point of Sale) started...');
OTT_GENERATED_IN_POS = query_template;
%map_attributes_to_weights step
attributes= cell(8,1);
attributes(1,1)= {'7704166087189'};
attributes(2,1)= {'kishor'};
attributes(3,1)= {'40202127'};
attributes(4,1)= {'santhakumari'};
```

```
attributes(5,1)= {'55145789'};
attributes(6,1)= {'manju'};
attributes(7,1)= {'keerthana'};
attributes(8,1)= {'kritha'};
disp('The captured attributes for the user follows')
disp(attributes)
disp('Performing map_attributes_to_weights')
max_attributes_len = 16;
for i=1:8
   padded_len = max_attributes_len  - length(attributes{i});
   attributes(i,1) = map_attributes_to_weights(attributes(i),padded_len);
end
disp(attributes)
str = strcat(attributes);
disp(str)
weight1 = dec2bin(str{1},1);weight2 = dec2bin(str{2},1);weight3 = dec2bin(str{3},1);weight4 = dec2bin(str{4},1);
weight5 = dec2bin(str{5},1);weight6 = dec2bin(str{6},1);weight7 = dec2bin(str{7},1);weight8 = dec2bin(str{8},1);
disp('map_attributes_to_weights completed...')
disp('weights are as follows:')
fprintf('weight1:%s\r\n',weight1)
fprintf('weight2:%s\r\n',weight2)
fprintf('weight3:%s\r\n',weight3)
fprintf('weight4:%s\r\n',weight4)
fprintf('weight5:%s\r\n',weight5)
fprintf('weight6:%s\r\n',weight6)
fprintf('weight7:%s\r\n',weight7)
fprintf('weight8:%s\r\n',weight8)
weights = {weight1,weight2,weight3,weight4,weight5,weight6,weight7,weight8};
disp(weights)
%derive Biometric Keys (BKs) step
BK_Temp = [];
for i=1:1000
   BK_Temp = [BK_Temp;sprintf('%s',weights{randperm(length(weights))})];
end
%map the 128 bits of BK_Temp to BK
BK= [];
for i=1:length(BK_Temp)
  BK =[BK;sprintf('%s', BK_Temp(i,1:128))];
end
disp('Biometric Keys of length 128 bits are derived successfully!!!')
disp('BKs are as follows')
for i=1:length(BK)
   disp(BK(i,:))
end
%Compute Biometric Template Transformation (BTT) of length 128 bytes
BK_Vault1= [];
BK_Vault2= [];
BK_Final = [];
for i=1:length(BK_Temp)
  BK_Vault1=[BK_Vault1;sprintf('%s', BK(i,1:16))];
  BK_Vault2=[BK_Vault2;sprintf('%s', BK(i,113:128))];
  BK_Final=[BK_Final;sprintf('%s', strcat(BK_Vault1(i,1:16),BK_Vault2(i,1:16)))];
end
disp('displaying sizeof BK_Final')
disp(size(BK_Final))
disp(length(BK_Final))

BK_Finally = [];
for i=1:1000
   BK_Finally=[BK_Finally;sprintf('%s', num2str(BK_Final(i,:)))];
   disp(BK_Final(i,:))
end
BK_Final_Int = [];
for i=1:1000
   BK_Final_Int=[BK_Final_Int;sprintf('%04i', bin2dec(BK_Finally(i,1:4)))];
   disp(BK_Final_Int(i,:))
end
 [x y] = size(query_template)
query _template_x = [];
query _template_y = [];
for i=1:x
   query _template_x=[ query _template_x;sprintf('%04i', query _template(i,1))];
   query _template_y=[ query _template_y;sprintf('%04i', query _template(i,2))];
end
TT = [];
```

```
disp('Minutiaes follows....')
for i=1:x
   disp([query _template_x(i,:) ',' query_template_y(i,:)])
end
j = 1;
for i=1:1000
   if(mod(i, x) == 0)
       j = 1;
   end
TT=[TT;sprintf('%012s',strcat(BK_Final_Int(i,:), query _template_x(j,:), query _y(j,:)))];
   j = j + 1;
end
disp('TT follows....')

for i=1:length(BK_Final)
   disp(TT(i,:))
end
%Generate IOTT by salting TTs and TSNs (Transaction Numbers)
disp('IOTT follows....')
IOTT = [];
for TSN=1:x
   IOTT=[IOTT;sprintf('%016s', strcat(BHC(TSN,:),num2str(TSN)))];
   disp(IOTT(TSN,:))
end
%Generate OTT by salting IOTT with DS (DateStamp) and TS (Time Stamp)
disp('OTT follows....')

OTT1 = [];
OTT = [];
TEMP_OTT_GENERATED_IN_POS = [];

S= datestr(now);
for DS_TS=1:x
   OTT1=[OTT1;sprintf('%032s', strcat(num2str(IOTT(DS_TS,:)),num2str(S)))];%Salting OTT with DS &TS
   %disp(OTT1(DS_TS,:));
   OTT=[OTT;sprintf('%i', OTT1(DS_TS,:))];
   %disp(OTT(DS_TS,:));
   TEMP_OTT_GENERATED_IN_ POS =[TEMP_OTT_GENERATED_IN_ POS;sprintf('%X', OTT(DS_TS,17:19))];
   %disp(TEMP_OTT_GENERATED_IN_ POS (DS_TS,:));
end
temp_template = [];
temp_template = TEMP_OTT_GENERATED_IN_ POS;
input_template = [];
input_template = OTT_GENERATED_IN_ POS;
OTT_GENERATED_IN_ POS = OTT_GENERATION(input_template, temp_template);
 end
```

Table 7.3: Matlab script for calculating FAR and FRR

```
%Script to generate FAR and FRR
function  build_far_frr( )
   load('db.mat');
   P=80
   far=zeros(100,P); frr=zeros(100,P);
   for p=1:P
      enrolled_template = ff{p};
      S=zeros(80,1);
      for i=1:P
          query_template = ff{i};
         OTT_GENERATED_IN_BAS = OTT_BAS(enrolled_template);
         OTT_GENERATED_IN_POS = OTT_POS(query_template);
         OTT_Enrolled = OTT_GENERATED_IN_BAS;
         OTT_Query = OTT_GENERATED_IN_POS;
         if(i>72)
            second=['1' num2str(fix((i-1)/8)+1) '_' num2str(mod(i-1,8)+1)];
         else
            second=['10' num2str(fix((i-1)/8)+1) '_' num2str(mod(i-1,8)+1)];
         end
         S(i)=match(OTT_Enrolled,OTT_Query);
         fprintf(['\b\b' num2str(i)]);
      end

      gt=[1 2 3 4 5 6 7 8]+(fix((p-1)/8)*8); k=0;
      for a=0.01:.01:1%values from 0.01 to 1 incremented by .01
```

```
        k=k+1; s=0;
        g=find(S>a);
        for i=1:length(gt)
            s=s+sum(g==gt(i));
        end
        far(k,p)=(length(g)-s)/72
        frr(k,p)=1-s/8;
      end
   end
   save('far.mat','far');
   save('frr.mat','frr');
end
```

Table 7.4 below illustartes the full test log obtained during the execution of test case1 in both CFAS and PFAS. Please refer to section 7.2.1.1 for the details of test case 1. Each line in the test log shows a matching or comparison score (threshold *t*) obtained after computing the similarity between two fingerprint images of the FVC2002-DB1_B database. The value of threshold *t* ranges between 0 and 1.

Table 7.4: Test log of Test case 1

CFAS Log:

Computing similarity between 101_1.tif and 101_2 from FVC2002 : 0.77067
Computing similarity between 101_1.tif and 101_3 from FVC2002 : 0.68476
Computing similarity between 101_1.tif and 101_4 from FVC2002 : 0.88396
Computing similarity between 101_1.tif and 101_5 from FVC2002 : 0.6742
Computing similarity between 101_1.tif and 101_6 from FVC2002 : 0.71263
Computing similarity between 101_1.tif and 101_7 from FVC2002 : 0.5766
Computing similarity between 101_1.tif and 101_8 from FVC2002 : 0.73983
Computing similarity between 101_2.tif and 101_3 from FVC2002 : 0.77005
Computing similarity between 101_2.tif and 101_4 from FVC2002 : 0.78881
Computing similarity between 101_2.tif and 101_5 from FVC2002 : 0.6532
Computing similarity between 101_2.tif and 101_6 from FVC2002 : 0.6233
Computing similarity between 101_2.tif and 101_7 from FVC2002 : 0.56737
Computing similarity between 101_2.tif and 101_8 from FVC2002 : 0.68516
Computing similarity between 101_3.tif and 101_4 from FVC2002 : 0.60075
Computing similarity between 101_3.tif and 101_5 from FVC2002 : 0.58038
Computing similarity between 101_3.tif and 101_6 from FVC2002 : 0.55382
Computing similarity between 101_3.tif and 101_7 from FVC2002 : 0.42656
Computing similarity between 101_3.tif and 101_8 from FVC2002 : 0.74927
Computing similarity between 101_4.tif and 101_5 from FVC2002 : 0.69007
Computing similarity between 101_4.tif and 101_6 from FVC2002 : 0.52679
Computing similarity between 101_4.tif and 101_7 from FVC2002 : 0.66394
Computing similarity between 101_4.tif and 101_8 from FVC2002 : 0.66815
Computing similarity between 101_5.tif and 101_6 from FVC2002 : 0.35233
Computing similarity between 101_5.tif and 101_7 from FVC2002 : 0.42762
Computing similarity between 101_5.tif and 101_8 from FVC2002 : 0.5164
Computing similarity between 101_6.tif and 101_7 from FVC2002 : 0.65915
Computing similarity between 101_6.tif and 101_8 from FVC2002 : 0.60648
Computing similarity between 101_7.tif and 101_8 from FVC2002 : 0.44854
Computing similarity between 102_1.tif and 102_2 from FVC2002 : 0.71028
Computing similarity between 102_1.tif and 102_3 from FVC2002 : 0.53875
Computing similarity between 102_1.tif and 102_4 from FVC2002 : 0.72056
Computing similarity between 102_1.tif and 102_5 from FVC2002 : 0.3118
Computing similarity between 102_1.tif and 102_6 from FVC2002 : 0.62655
Computing similarity between 102_1.tif and 102_7 from FVC2002 : 0.54697
Computing similarity between 102_1.tif and 102_8 from FVC2002 : 0.54697
Computing similarity between 102_2.tif and 102_3 from FVC2002 : 0.60987
Computing similarity between 102_2.tif and 102_4 from FVC2002 : 0.79723
Computing similarity between 102_2.tif and 102_5 from FVC2002 : 0.42712
Computing similarity between 102_2.tif and 102_6 from FVC2002 : 0.69322
Computing similarity between 102_2.tif and 102_7 from FVC2002 : 0.65131
Computing similarity between 102_2.tif and 102_8 from FVC2002 : 0.68559
Computing similarity between 102_3.tif and 102_4 from FVC2002 : 0.70548
Computing similarity between 102_3.tif and 102_5 from FVC2002 : 0.50395
Computing similarity between 102_3.tif and 102_6 from FVC2002 : 0.47712
Computing similarity between 102_3.tif and 102_7 from FVC2002 : 0.68252

Computing similarity between 102_3.tif and 102_8 from FVC2002 : 0.77353
Computing similarity between 102_4.tif and 102_5 from FVC2002 : 0.41478
Computing similarity between 102_4.tif and 102_6 from FVC2002 : 0.70124
Computing similarity between 102_4.tif and 102_7 from FVC2002 : 0.74901
Computing similarity between 102_4.tif and 102_8 from FVC2002 : 0.5243
Computing similarity between 102_5.tif and 102_6 from FVC2002 : 0.22542
Computing similarity between 102_5.tif and 102_7 from FVC2002 : 0.30096
Computing similarity between 102_5.tif and 102_8 from FVC2002 : 0.48154
Computing similarity between 102_6.tif and 102_7 from FVC2002 : 0.78155
Computing similarity between 102_6.tif and 102_8 from FVC2002 : 0.48847
Computing similarity between 102_7.tif and 102_8 from FVC2002 : 0.6087
Computing similarity between 103_1.tif and 103_2 from FVC2002 : 0.89709
Computing similarity between 103_1.tif and 103_3 from FVC2002 : 0.56393
Computing similarity between 103_1.tif and 103_4 from FVC2002 : 0.48795
Computing similarity between 103_1.tif and 103_5 from FVC2002 : 0.5409
Computing similarity between 103_1.tif and 103_6 from FVC2002 : 0.61721
Computing similarity between 103_1.tif and 103_7 from FVC2002 : 0.75897
Computing similarity between 103_1.tif and 103_8 from FVC2002 : 0.57394
Computing similarity between 103_2.tif and 103_3 from FVC2002 : 0.46819
Computing similarity between 103_2.tif and 103_4 from FVC2002 : 0.53033
Computing similarity between 103_2.tif and 103_5 from FVC2002 : 0.44907
Computing similarity between 103_2.tif and 103_6 from FVC2002 : 0.55902
Computing similarity between 103_2.tif and 103_7 from FVC2002 : 0.73324
Computing similarity between 103_2.tif and 103_8 from FVC2002 : 0.44557
Computing similarity between 103_3.tif and 103_4 from FVC2002 : 0.36116
Computing similarity between 103_3.tif and 103_5 from FVC2002 : 0.54214
Computing similarity between 103_3.tif and 103_6 from FVC2002 : 0.57104
Computing similarity between 103_3.tif and 103_7 from FVC2002 : 0.56175
Computing similarity between 103_3.tif and 103_8 from FVC2002 : 0.60687
Computing similarity between 103_4.tif and 103_5 from FVC2002 : 0.23094
Computing similarity between 103_4.tif and 103_6 from FVC2002 : 0.36893
Computing similarity between 103_4.tif and 103_7 from FVC2002 : 0.62217
Computing similarity between 103_4.tif and 103_8 from FVC2002 : 0.2100
Computing similarity between 103_5.tif and 103_6 from FVC2002 : 0.54772
Computing similarity between 103_5.tif and 103_7 from FVC2002 : 0.50289
Computing similarity between 103_5.tif and 103_8 from FVC2002 : 0.53358
Computing similarity between 103_6.tif and 103_7 from FVC2002 : 0.52466
Computing similarity between 103_6.tif and 103_8 from FVC2002 : 0.48709
Computing similarity between 103_7.tif and 103_8 from FVC2002 : 0.52273
Computing similarity between 104_1.tif and 104_2 from FVC2002 : 0.77977
Computing similarity between 104_1.tif and 104_3 from FVC2002 : 0.86772
Computing similarity between 104_1.tif and 104_4 from FVC2002 : 0.6228
Computing similarity between 104_1.tif and 104_5 from FVC2002 : 0.66421
Computing similarity between 104_1.tif and 104_6 from FVC2002 : 0.5003
Computing similarity between 104_1.tif and 104_7 from FVC2002 : 0.67414
Computing similarity between 104_1.tif and 104_8 from FVC2002 : 0.92272
Computing similarity between 104_2.tif and 104_3 from FVC2002 : 0.8274
Computing similarity between 104_2.tif and 104_4 from FVC2002 : 0.87099
Computing similarity between 104_2.tif and 104_5 from FVC2002 : 0.7206
Computing similarity between 104_2.tif and 104_6 from FVC2002 : 0.59658
Computing similarity between 104_2.tif and 104_7 from FVC2002 : 0.77916
Computing similarity between 104_2.tif and 104_8 from FVC2002 : 0.78527
Computing similarity between 104_3.tif and 104_4 from FVC2002 : 0.69663
Computing similarity between 104_3.tif and 104_5 from FVC2002 : 0.67241
Computing similarity between 104_3.tif and 104_6 from FVC2002 : 0.50522
Computing similarity between 104_3.tif and 104_7 from FVC2002 : 0.69797
Computing similarity between 104_3.tif and 104_8 from FVC2002 : 0.85786
Computing similarity between 104_4.tif and 104_5 from FVC2002 : 0.74978
Computing similarity between 104_4.tif and 104_6 from FVC2002 : 0.52378
Computing similarity between 104_4.tif and 104_7 from FVC2002 : 0.66198
Computing similarity between 104_4.tif and 104_8 from FVC2002 : 0.61572
Computing similarity between 104_5.tif and 104_6 from FVC2002 : 0.55002
Computing similarity between 104_5.tif and 104_7 from FVC2002 : 0.63095
Computing similarity between 104_5.tif and 104_8 from FVC2002 : 0.60038
Computing similarity between 104_6.tif and 104_7 from FVC2002 : 0.72952
Computing similarity between 104_6.tif and 104_8 from FVC2002 : 0.50679
Computing similarity between 104_7.tif and 104_8 from FVC2002 : 0.69244
Computing similarity between 105_1.tif and 105_2 from FVC2002 : 0.7907
Computing similarity between 105_1.tif and 105_3 from FVC2002 : 0.81395
Computing similarity between 105_1.tif and 105_4 from FVC2002 : 0.60521
Computing similarity between 105_1.tif and 105_5 from FVC2002 : 0.55748
Computing similarity between 105_1.tif and 105_6 from FVC2002 : 0.74709
Computing similarity between 105_1.tif and 105_7 from FVC2002 : 0.59287
Computing similarity between 105_1.tif and 105_8 from FVC2002 : 0.623
Computing similarity between 105_2.tif and 105_3 from FVC2002 : 0.95349

Computing similarity between 105_2.tif and 105_4 from FVC2002 : 0.74931
Computing similarity between 105_2.tif and 105_5 from FVC2002 : 0.61057
Computing similarity between 105_2.tif and 105_6 from FVC2002 : 0.77822
Computing similarity between 105_2.tif and 105_7 from FVC2002 : 0.69598
Computing similarity between 105_2.tif and 105_8 from FVC2002 : 0.76459
Computing similarity between 105_3.tif and 105_4 from FVC2002 : 0.72049
Computing similarity between 105_3.tif and 105_5 from FVC2002 : 0.63712
Computing similarity between 105_3.tif and 105_6 from FVC2002 : 0.77822
Computing similarity between 105_3.tif and 105_7 from FVC2002 : 0.72175
Computing similarity between 105_3.tif and 105_8 from FVC2002 : 0.79291
Computing similarity between 105_4.tif and 105_5 from FVC2002 : 0.62505
Computing similarity between 105_4.tif and 105_6 from FVC2002 : 0.50149
Computing similarity between 105_4.tif and 105_7 from FVC2002 : 0.57499
Computing similarity between 105_4.tif and 105_8 from FVC2002 : 0.70186
Computing similarity between 105_5.tif and 105_6 from FVC2002 : 0.49747
Computing similarity between 105_5.tif and 105_7 from FVC2002 : 0.73561
Computing similarity between 105_5.tif and 105_8 from FVC2002 : 0.67883
Computing similarity between 105_6.tif and 105_7 from FVC2002 : 0.41404
Computing similarity between 105_6.tif and 105_8 from FVC2002 : 0.45486
Computing similarity between 105_7.tif and 105_8 from FVC2002 : 0.65915
Computing similarity between 106_1.tif and 106_2 from FVC2002 : 0.86275
Computing similarity between 106_1.tif and 106_3 from FVC2002 : 0.52394
Computing similarity between 106_1.tif and 106_4 from FVC2002 : 0.29643
Computing similarity between 106_1.tif and 106_5 from FVC2002 : 0.52947
Computing similarity between 106_1.tif and 106_6 from FVC2002 : 0.52947
Computing similarity between 106_1.tif and 106_7 from FVC2002 : 0.83101
Computing similarity between 106_1.tif and 106_8 from FVC2002 : 0.62406
Computing similarity between 106_2.tif and 106_3 from FVC2002 : 0.59878
Computing similarity between 106_2.tif and 106_4 from FVC2002 : 0.37728
Computing similarity between 106_2.tif and 106_5 from FVC2002 : 0.59853
Computing similarity between 106_2.tif and 106_6 from FVC2002 : 0.64457
Computing similarity between 106_2.tif and 106_7 from FVC2002 : 0.83101
Computing similarity between 106_2.tif and 106_8 from FVC2002 : 0.65006
Computing similarity between 106_3.tif and 106_4 from FVC2002 : 0.56578
Computing similarity between 106_3.tif and 106_5 from FVC2002 : 0.63709
Computing similarity between 106_3.tif and 106_6 from FVC2002 : 0.63709
Computing similarity between 106_3.tif and 106_7 from FVC2002 : 0.62609
Computing similarity between 106_3.tif and 106_8 from FVC2002 : 0.47148
Computing similarity between 106_4.tif and 106_5 from FVC2002 : 0.63277
Computing similarity between 106_4.tif and 106_6 from FVC2002 : 0.60113
Computing similarity between 106_4.tif and 106_7 from FVC2002 : 0.36067
Computing similarity between 106_4.tif and 106_8 from FVC2002 : 0.39311
Computing similarity between 106_5.tif and 106_6 from FVC2002 : 0.78378
Computing similarity between 106_5.tif and 106_7 from FVC2002 : 0.6162
Computing similarity between 106_5.tif and 106_8 from FVC2002 : 0.70215
Computing similarity between 106_6.tif and 106_7 from FVC2002 : 0.6162
Computing similarity between 106_6.tif and 106_8 from FVC2002 : 0.67162
Computing similarity between 106_7.tif and 106_8 from FVC2002 : 0.78302
Computing similarity between 107_1.tif and 107_2 from FVC2002 : 0.8428
Computing similarity between 107_1.tif and 107_3 from FVC2002 : 0.78179
Computing similarity between 107_1.tif and 107_4 from FVC2002 : 0.80812
Computing similarity between 107_1.tif and 107_5 from FVC2002 : 0.63475
Computing similarity between 107_1.tif and 107_6 from FVC2002 : 0.59761
Computing similarity between 107_1.tif and 107_7 from FVC2002 : 0.65161
Computing similarity between 107_1.tif and 107_8 from FVC2002 : 0.78467
Computing similarity between 107_2.tif and 107_3 from FVC2002 : 0.82733
Computing similarity between 107_2.tif and 107_4 from FVC2002 : 0.77502
Computing similarity between 107_2.tif and 107_5 from FVC2002 : 0.71458
Computing similarity between 107_2.tif and 107_6 from FVC2002 : 0.58356
Computing similarity between 107_2.tif and 107_7 from FVC2002 : 0.70294
Computing similarity between 107_2.tif and 107_8 from FVC2002 : 0.78378
Computing similarity between 107_3.tif and 107_4 from FVC2002 : 0.76249
Computing similarity between 107_3.tif and 107_5 from FVC2002 : 0.72049
Computing similarity between 107_3.tif and 107_6 from FVC2002 : 0.54132
Computing similarity between 107_3.tif and 107_7 from FVC2002 : 0.69703
Computing similarity between 107_3.tif and 107_8 from FVC2002 : 0.80226
Computing similarity between 107_4.tif and 107_5 from FVC2002 : 0.62094
Computing similarity between 107_4.tif and 107_6 from FVC2002 : 0.55539
Computing similarity between 107_4.tif and 107_7 from FVC2002 : 0.67402
Computing similarity between 107_4.tif and 107_8 from FVC2002 : 0.84548
Computing similarity between 107_5.tif and 107_6 from FVC2002 : 0.57499
Computing similarity between 107_5.tif and 107_7 from FVC2002 : 0.6966
Computing similarity between 107_5.tif and 107_8 from FVC2002 : 0.74564
Computing similarity between 107_6.tif and 107_7 from FVC2002 : 0.54829
Computing similarity between 107_6.tif and 107_8 from FVC2002 : 0.50019

Computing similarity between 107_7.tif and 107_8 from FVC2002 : 0.75142
Computing similarity between 108_1.tif and 108_2 from FVC2002 : 0.7122
Computing similarity between 108_1.tif and 108_3 from FVC2002 : 0.86032
Computing similarity between 108_1.tif and 108_4 from FVC2002 : 0.67541
Computing similarity between 108_1.tif and 108_5 from FVC2002 : 0.76345
Computing similarity between 108_1.tif and 108_6 from FVC2002 : 0.71931
Computing similarity between 108_1.tif and 108_7 from FVC2002 : 0.74627
Computing similarity between 108_1.tif and 108_8 from FVC2002 : 0.81484
Computing similarity between 108_2.tif and 108_3 from FVC2002 : 0.69876
Computing similarity between 108_2.tif and 108_4 from FVC2002 : 0.55192
Computing similarity between 108_2.tif and 108_5 from FVC2002 : 0.7751
Computing similarity between 108_2.tif and 108_6 from FVC2002 : 0.6039
Computing similarity between 108_2.tif and 108_7 from FVC2002 : 0.62128
Computing similarity between 108_2.tif and 108_8 from FVC2002 : 0.67686
Computing similarity between 108_3.tif and 108_4 from FVC2002 : 0.74237
Computing similarity between 108_3.tif and 108_5 from FVC2002 : 0.76677
Computing similarity between 108_3.tif and 108_6 from FVC2002 : 0.73936
Computing similarity between 108_3.tif and 108_7 from FVC2002 : 0.71954
Computing similarity between 108_3.tif and 108_8 from FVC2002 : 0.76081
Computing similarity between 108_4.tif and 108_5 from FVC2002 : 0.63197
Computing similarity between 108_4.tif and 108_6 from FVC2002 : 0.59544
Computing similarity between 108_4.tif and 108_7 from FVC2002 : 0.67541
Computing similarity between 108_4.tif and 108_8 from FVC2002 : 0.68677
Computing similarity between 108_5.tif and 108_6 from FVC2002 : 0.69283
Computing similarity between 108_5.tif and 108_7 from FVC2002 : 0.66801
Computing similarity between 108_5.tif and 108_8 from FVC2002 : 0.75781
Computing similarity between 108_6.tif and 108_7 from FVC2002 : 0.71931
Computing similarity between 108_6.tif and 108_8 from FVC2002 : 0.79639
Computing similarity between 108_7.tif and 108_8 from FVC2002 : 0.77039
Computing similarity between 109_1.tif and 109_2 from FVC2002 : 0.69102
Computing similarity between 109_1.tif and 109_3 from FVC2002 : 0.83519
Computing similarity between 109_1.tif and 109_4 from FVC2002 : 0.67857
Computing similarity between 109_1.tif and 109_5 from FVC2002 : 0.77205
Computing similarity between 109_1.tif and 109_6 from FVC2002 : 0.41404
Computing similarity between 109_1.tif and 109_7 from FVC2002 : 0.63168
Computing similarity between 109_1.tif and 109_8 from FVC2002 : 0.79358
Computing similarity between 109_2.tif and 109_3 from FVC2002 : 0.64639
Computing similarity between 109_2.tif and 109_4 from FVC2002 : 0.76376
Computing similarity between 109_2.tif and 109_5 from FVC2002 : 0.75048
Computing similarity between 109_2.tif and 109_6 from FVC2002 : 0.28109
Computing similarity between 109_2.tif and 109_7 from FVC2002 : 0.64327
Computing similarity between 109_2.tif and 109_8 from FVC2002 : 0.70273
Computing similarity between 109_3.tif and 109_4 from FVC2002 : 0.73497
Computing similarity between 109_3.tif and 109_5 from FVC2002 : 0.75501
Computing similarity between 109_3.tif and 109_6 from FVC2002 : 0.48412
Computing similarity between 109_3.tif and 109_7 from FVC2002 : 0.65653
Computing similarity between 109_3.tif and 109_8 from FVC2002 : 0.80687
Computing similarity between 109_4.tif and 109_5 from FVC2002 : 0.84223
Computing similarity between 109_4.tif and 109_6 from FVC2002 : 0.44854
Computing similarity between 109_4.tif and 109_7 from FVC2002 : 0.59658
Computing similarity between 109_4.tif and 109_8 from FVC2002 : 0.75907
Computing similarity between 109_5.tif and 109_6 from FVC2002 : 0.44074
Computing similarity between 109_5.tif and 109_7 from FVC2002 : 0.65517
Computing similarity between 109_5.tif and 109_8 from FVC2002 : 0.71197
Computing similarity between 109_6.tif and 109_7 from FVC2002 : 0.37293
Computing similarity between 109_6.tif and 109_8 from FVC2002 : 0.33333
Computing similarity between 109_7.tif and 109_8 from FVC2002 : 0.67806
Computing similarity between 110_1.tif and 110_2 from FVC2002 : 0.81041
Computing similarity between 110_1.tif and 110_3 from FVC2002 : 0.7999
Computing similarity between 110_1.tif and 110_4 from FVC2002 : 0.43593
Computing similarity between 110_1.tif and 110_5 from FVC2002 : 0.39687
Computing similarity between 110_1.tif and 110_6 from FVC2002 : 0.574
Computing similarity between 110_1.tif and 110_7 from FVC2002 : 0.39472
Computing similarity between 110_1.tif and 110_8 from FVC2002 : 0.42291
Computing similarity between 110_2.tif and 110_3 from FVC2002 : 0.78065
Computing similarity between 110_2.tif and 110_4 from FVC2002 : 0.5164
Computing similarity between 110_2.tif and 110_5 from FVC2002 : 0.47464
Computing similarity between 110_2.tif and 110_6 from FVC2002 : 0.55777
Computing similarity between 110_2.tif and 110_7 from FVC2002 : 0.43836
Computing similarity between 110_2.tif and 110_8 from FVC2002 : 0.50098
Computing similarity between 110_3.tif and 110_4 from FVC2002 : 0.59948
Computing similarity between 110_3.tif and 110_5 from FVC2002 : 0.60234
Computing similarity between 110_3.tif and 110_6 from FVC2002 : 0.51479
Computing similarity between 110_3.tif and 110_7 from FVC2002 : 0.58158
Computing similarity between 110_3.tif and 110_8 from FVC2002 : 0.60687

Computing similarity between 110_4.tif and 110_5 from FVC2002 : 0.82067
Computing similarity between 110_4.tif and 110_6 from FVC2002 : 0.42433
Computing similarity between 110_4.tif and 110_7 from FVC2002 : 0.66697
Computing similarity between 110_4.tif and 110_8 from FVC2002 : 0.84887
Computing similarity between 110_5.tif and 110_6 from FVC2002 : 0.28365
Computing similarity between 110_5.tif and 110_7 from FVC2002 : 0.85986
Computing similarity between 110_5.tif and 110_8 from FVC2002 : 0.82801
Computing similarity between 110_6.tif and 110_7 from FVC2002 : 0.29939
Computing similarity between 110_6.tif and 110_8 from FVC2002 : 0.33682
Computing similarity between 110_7.tif and 110_8 from FVC2002 : 0.79412

PFAS Log:
Computing similarity between 101_1.tif and 101_2 from FVC2002 : 0.7298
Computing similarity between 101_1.tif and 101_3 from FVC2002 : 0.65275
Computing similarity between 101_1.tif and 101_4 from FVC2002 : 0.81903
Computing similarity between 101_1.tif and 101_5 from FVC2002 : 0.62869
Computing similarity between 101_1.tif and 101_6 from FVC2002 : 0.5808
Computing similarity between 101_1.tif and 101_7 from FVC2002 : 0.59079
Computing similarity between 101_1.tif and 101_8 from FVC2002 : 0.66725
Computing similarity between 101_2.tif and 101_3 from FVC2002 : 0.72672
Computing similarity between 101_2.tif and 101_4 from FVC2002 : 0.70921
Computing similarity between 101_2.tif and 101_5 from FVC2002 : 0.6784
Computing similarity between 101_2.tif and 101_6 from FVC2002 : 0.41781
Computing similarity between 101_2.tif and 101_7 from FVC2002 : 0.54167
Computing similarity between 101_2.tif and 101_8 from FVC2002 : 0.65
Computing similarity between 101_3.tif and 101_4 from FVC2002 : 0.53675
Computing similarity between 101_3.tif and 101_5 from FVC2002 : 0.60678
Computing similarity between 101_3.tif and 101_6 from FVC2002 : 0.45675
Computing similarity between 101_3.tif and 101_7 from FVC2002 : 0.44721
Computing similarity between 101_3.tif and 101_8 from FVC2002 : 0.76026
Computing similarity between 101_4.tif and 101_5 from FVC2002 : 0.52636
Computing similarity between 101_4.tif and 101_6 from FVC2002 : 0.48626
Computing similarity between 101_4.tif and 101_7 from FVC2002 : 0.61828
Computing similarity between 101_4.tif and 101_8 from FVC2002 : 0.52372
Computing similarity between 101_5.tif and 101_6 from FVC2002 : 0.39192
Computing similarity between 101_5.tif and 101_7 from FVC2002 : 0.40202
Computing similarity between 101_5.tif and 101_8 from FVC2002 : 0.54272
Computing similarity between 101_6.tif and 101_7 from FVC2002 : 0.64993
Computing similarity between 101_6.tif and 101_8 from FVC2002 : 0.51995
Computing similarity between 101_7.tif and 101_8 from FVC2002 : 0.4
Computing similarity between 102_1.tif and 102_2 from FVC2002 : 0.6432
Computing similarity between 102_1.tif and 102_3 from FVC2002 : 0.49917
Computing similarity between 102_1.tif and 102_4 from FVC2002 : 0.647
Computing similarity between 102_1.tif and 102_5 from FVC2002 : 0.26414
Computing similarity between 102_1.tif and 102_6 from FVC2002 : 0.54777
Computing similarity between 102_1.tif and 102_7 from FVC2002 : 0.52919
Computing similarity between 102_1.tif and 102_8 from FVC2002 : 0.41338
Computing similarity between 102_2.tif and 102_3 from FVC2002 : 0.53099
Computing similarity between 102_2.tif and 102_4 from FVC2002 : 0.71692
Computing similarity between 102_2.tif and 102_5 from FVC2002 : 0.32781
Computing similarity between 102_2.tif and 102_6 from FVC2002 : 0.5827
Computing similarity between 102_2.tif and 102_7 from FVC2002 : 0.62915
Computing similarity between 102_2.tif and 102_8 from FVC2002 : 0.67651
Computing similarity between 102_3.tif and 102_4 from FVC2002 : 0.57864
Computing similarity between 102_3.tif and 102_5 from FVC2002 : 0.37796
Computing similarity between 102_3.tif and 102_6 from FVC2002 : 0.5112
Computing similarity between 102_3.tif and 102_7 from FVC2002 : 0.57907
Computing similarity between 102_3.tif and 102_8 from FVC2002 : 0.72803
Computing similarity between 102_4.tif and 102_5 from FVC2002 : 0.35722
Computing similarity between 102_4.tif and 102_6 from FVC2002 : 0.55216
Computing similarity between 102_4.tif and 102_7 from FVC2002 : 0.72169
Computing similarity between 102_4.tif and 102_8 from FVC2002 : 0.47919
Computing similarity between 102_5.tif and 102_6 from FVC2002 : 0.18033
Computing similarity between 102_5.tif and 102_7 from FVC2002 : 0.29463
Computing similarity between 102_5.tif and 102_8 from FVC2002 : 0.48154
Computing similarity between 102_6.tif and 102_7 from FVC2002 : 0.73321
Computing similarity between 102_6.tif and 102_8 from FVC2002 : 0.4559
Computing similarity between 102_7.tif and 102_8 from FVC2002 : 0.55332
Computing similarity between 103_1.tif and 103_2 from FVC2002 : 0.81135
Computing similarity between 103_1.tif and 103_3 from FVC2002 : 0.52868
Computing similarity between 103_1.tif and 103_4 from FVC2002 : 0.48795
Computing similarity between 103_1.tif and 103_5 from FVC2002 : 0.43948
Computing similarity between 103_1.tif and 103_6 from FVC2002 : 0.54646
Computing similarity between 103_1.tif and 103_7 from FVC2002 : 0.69825
Computing similarity between 103_1.tif and 103_8 from FVC2002 : 0.45096

Computing similarity between 103_2.tif and 103_3 from FVC2002 : 0.41703
Computing similarity between 103_2.tif and 103_4 from FVC2002 : 0.46188
Computing similarity between 103_2.tif and 103_5 from FVC2002 : 0.4
Computing similarity between 103_2.tif and 103_6 from FVC2002 : 0.50289
Computing similarity between 103_2.tif and 103_7 from FVC2002 : 0.6825
Computing similarity between 103_2.tif and 103_8 from FVC2002 : 0.38806
Computing similarity between 103_3.tif and 103_4 from FVC2002 : 0.36116
Computing similarity between 103_3.tif and 103_5 from FVC2002 : 0.45873
Computing similarity between 103_3.tif and 103_6 from FVC2002 : 0.48685
Computing similarity between 103_3.tif and 103_7 from FVC2002 : 0.48685
Computing similarity between 103_3.tif and 103_8 from FVC2002 : 0.60687
Computing similarity between 103_4.tif and 103_5 from FVC2002 : 0.23094
Computing similarity between 103_4.tif and 103_6 from FVC2002 : 0.36293
Computing similarity between 103_4.tif and 103_7 from FVC2002 : 0.46663
Computing similarity between 103_4.tif and 103_8 from FVC2002 : 0.21004
Computing similarity between 103_5.tif and 103_6 from FVC2002 : 0.53882
Computing similarity between 103_5.tif and 103_7 from FVC2002 : 0.46697
Computing similarity between 103_5.tif and 103_8 from FVC2002 : 0.53358
Computing similarity between 103_6.tif and 103_7 from FVC2002 : 0.45161
Computing similarity between 103_6.tif and 103_8 from FVC2002 : 0.43561
Computing similarity between 103_7.tif and 103_8 from FVC2002 : 0.43561
Computing similarity between 104_1.tif and 104_2 from FVC2002 : 0.67248
Computing similarity between 104_1.tif and 104_3 from FVC2002 : 0.77094
Computing similarity between 104_1.tif and 104_4 from FVC2002 : 0.61767
Computing similarity between 104_1.tif and 104_5 from FVC2002 : 0.57091
Computing similarity between 104_1.tif and 104_6 from FVC2002 : 0.46196
Computing similarity between 104_1.tif and 104_7 from FVC2002 : 0.65339
Computing similarity between 104_1.tif and 104_8 from FVC2002 : 0.83162
Computing similarity between 104_2.tif and 104_3 from FVC2002 : 0.75385
Computing similarity between 104_2.tif and 104_4 from FVC2002 : 0.8116
Computing similarity between 104_2.tif and 104_5 from FVC2002 : 0.65305
Computing similarity between 104_2.tif and 104_6 from FVC2002 : 0.54394
Computing similarity between 104_2.tif and 104_7 from FVC2002 : 0.68566
Computing similarity between 104_2.tif and 104_8 from FVC2002 : 0.68229
Computing similarity between 104_3.tif and 104_4 from FVC2002 : 0.6333
Computing similarity between 104_3.tif and 104_5 from FVC2002 : 0.69642
Computing similarity between 104_3.tif and 104_6 from FVC2002 : 0.48651
Computing similarity between 104_3.tif and 104_7 from FVC2002 : 0.63149
Computing similarity between 104_3.tif and 104_8 from FVC2002 : 0.78824
Computing similarity between 104_4.tif and 104_5 from FVC2002 : 0.74978
Computing similarity between 104_4.tif and 104_6 from FVC2002 : 0.48349
Computing similarity between 104_4.tif and 104_7 from FVC2002 : 0.59042
Computing similarity between 104_4.tif and 104_8 from FVC2002 : 0.60927
Computing similarity between 104_5.tif and 104_6 from FVC2002 : 0.5271
Computing similarity between 104_5.tif and 104_7 from FVC2002 : 0.56989
Computing similarity between 104_5.tif and 104_8 from FVC2002 : 0.54458
Computing similarity between 104_6.tif and 104_7 from FVC2002 : 0.71366
Computing similarity between 104_6.tif and 104_8 from FVC2002 : 0.4822
Computing similarity between 104_7.tif and 104_8 from FVC2002 : 0.6338
Computing similarity between 105_1.tif and 105_2 from FVC2002 : 0.75
Computing similarity between 105_1.tif and 105_3 from FVC2002 : 0.81818
Computing similarity between 105_1.tif and 105_4 from FVC2002 : 0.51282
Computing similarity between 105_1.tif and 105_5 from FVC2002 : 0.47238
Computing similarity between 105_1.tif and 105_6 from FVC2002 : 0.64623
Computing similarity between 105_1.tif and 105_7 from FVC2002 : 0.55277
Computing similarity between 105_1.tif and 105_8 from FVC2002 : 0.52296
Computing similarity between 105_2.tif and 105_3 from FVC2002 : 0.93182
Computing similarity between 105_2.tif and 105_4 from FVC2002 : 0.74074
Computing similarity between 105_2.tif and 105_5 from FVC2002 : 0.55111
Computing similarity between 105_2.tif and 105_6 from FVC2002 : 0.677
Computing similarity between 105_2.tif and 105_7 from FVC2002 : 0.6784
Computing similarity between 105_2.tif and 105_8 from FVC2002 : 0.66058
Computing similarity between 105_3.tif and 105_4 from FVC2002 : 0.74074
Computing similarity between 105_3.tif and 105_5 from FVC2002 : 0.60359
Computing similarity between 105_3.tif and 105_6 from FVC2002 : 0.70778
Computing similarity between 105_3.tif and 105_7 from FVC2002 : 0.70353
Computing similarity between 105_3.tif and 105_8 from FVC2002 : 0.71563
Computing similarity between 105_4.tif and 105_5 from FVC2002 : 0.55926
Computing similarity between 105_4.tif and 105_6 from FVC2002 : 0.50149
Computing similarity between 105_4.tif and 105_7 from FVC2002 : 0.53545
Computing similarity between 105_4.tif and 105_8 from FVC2002 : 0.62106
Computing similarity between 105_5.tif and 105_6 from FVC2002 : 0.46193
Computing similarity between 105_5.tif and 105_7 from FVC2002 : 0.72532
Computing similarity between 105_5.tif and 105_8 from FVC2002 : 0.66742
Computing similarity between 105_6.tif and 105_7 from FVC2002 : 0.30619

Computing similarity between 105_6.tif and 105_8 from FVC2002 : 0.33541
Computing similarity between 105_7.tif and 105_8 from FVC2002 : 0.63901
Computing similarity between 106_1.tif and 106_2 from FVC2002 : 0.88462
Computing similarity between 106_1.tif and 106_3 from FVC2002 : 0.50034
Computing similarity between 106_1.tif and 106_4 from FVC2002 : 0.29357
Computing similarity between 106_1.tif and 106_5 from FVC2002 : 0.52435
Computing similarity between 106_1.tif and 106_6 from FVC2002 : 0.50156
Computing similarity between 106_1.tif and 106_7 from FVC2002 : 0.81312
Computing similarity between 106_1.tif and 106_8 from FVC2002 : 0.60764
Computing similarity between 106_2.tif and 106_3 from FVC2002 : 0.55594
Computing similarity between 106_2.tif and 106_4 from FVC2002 : 0.34694
Computing similarity between 106_2.tif and 106_5 from FVC2002 : 0.59275
Computing similarity between 106_2.tif and 106_6 from FVC2002 : 0.56995
Computing similarity between 106_2.tif and 106_7 from FVC2002 : 0.85592
Computing similarity between 106_2.tif and 106_8 from FVC2002 : 0.63296
Computing similarity between 106_3.tif and 106_4 from FVC2002 : 0.51434
Computing similarity between 106_3.tif and 106_5 from FVC2002 : 0.63709
Computing similarity between 106_3.tif and 106_6 from FVC2002 : 0.52725
Computing similarity between 106_3.tif and 106_7 from FVC2002 : 0.61859
Computing similarity between 106_3.tif and 106_8 from FVC2002 : 0.53675
Computing similarity between 106_4.tif and 106_5 from FVC2002 : 0.60113
Computing similarity between 106_4.tif and 106_6 from FVC2002 : 0.56949
Computing similarity between 106_4.tif and 106_7 from FVC2002 : 0.32665
Computing similarity between 106_4.tif and 106_8 from FVC2002 : 0.35136
Computing similarity between 106_5.tif and 106_6 from FVC2002 : 0.78378
Computing similarity between 106_5.tif and 106_7 from FVC2002 : 0.58345
Computing similarity between 106_5.tif and 106_8 from FVC2002 : 0.6003
Computing similarity between 106_6.tif and 106_7 from FVC2002 : 0.50735
Computing similarity between 106_6.tif and 106_8 from FVC2002 : 0.54027
Computing similarity between 106_7.tif and 106_8 from FVC2002 : 0.78881
Computing similarity between 107_1.tif and 107_2 from FVC2002 : 0.84717
Computing similarity between 107_1.tif and 107_3 from FVC2002 : 0.76105
Computing similarity between 107_1.tif and 107_4 from FVC2002 : 0.73855
Computing similarity between 107_1.tif and 107_5 from FVC2002 : 0.62505
Computing similarity between 107_1.tif and 107_6 from FVC2002 : 0.50022
Computing similarity between 107_1.tif and 107_7 from FVC2002 : 0.61599
Computing similarity between 107_1.tif and 107_8 from FVC2002 : 0.73422
Computing similarity between 107_2.tif and 107_3 from FVC2002 : 0.80704
Computing similarity between 107_2.tif and 107_4 from FVC2002 : 0.71119
Computing similarity between 107_2.tif and 107_5 from FVC2002 : 0.67445
Computing similarity between 107_2.tif and 107_6 from FVC2002 : 0.52099
Computing similarity between 107_2.tif and 107_7 from FVC2002 : 0.74147
Computing similarity between 107_2.tif and 107_8 from FVC2002 : 0.71053
Computing similarity between 107_3.tif and 107_4 from FVC2002 : 0.6396
Computing similarity between 107_3.tif and 107_5 from FVC2002 : 0.71225
Computing similarity between 107_3.tif and 107_6 from FVC2002 : 0.45868
Computing similarity between 107_3.tif and 107_7 from FVC2002 : 0.66683
Computing similarity between 107_3.tif and 107_8 from FVC2002 : 0.73367
Computing similarity between 107_4.tif and 107_5 from FVC2002 : 0.56125
Computing similarity between 107_4.tif and 107_6 from FVC2002 : 0.502
Computing similarity between 107_4.tif and 107_7 from FVC2002 : 0.60469
Computing similarity between 107_4.tif and 107_8 from FVC2002 : 0.64236
Computing similarity between 107_5.tif and 107_6 from FVC2002 : 0.5111
Computing similarity between 107_5.tif and 107_7 from FVC2002 : 0.6966
Computing similarity between 107_5.tif and 107_8 from FVC2002 : 0.6438
Computing similarity between 107_6.tif and 107_7 from FVC2002 : 0.52337
Computing similarity between 107_6.tif and 107_8 from FVC2002 : 0.41131
Computing similarity between 107_7.tif and 107_8 from FVC2002 : 0.64579
Computing similarity between 108_1.tif and 108_2 from FVC2002 : 0.63644
Computing similarity between 108_1.tif and 108_3 from FVC2002 : 0.79775
Computing similarity between 108_1.tif and 108_4 from FVC2002 : 0.60951
Computing similarity between 108_1.tif and 108_5 from FVC2002 : 0.58849
Computing similarity between 108_1.tif and 108_6 from FVC2002 : 0.64602
Computing similarity between 108_1.tif and 108_7 from FVC2002 : 0.72595
Computing similarity between 108_1.tif and 108_8 from FVC2002 : 0.78521
Computing similarity between 108_2.tif and 108_3 from FVC2002 : 0.68288
Computing similarity between 108_2.tif and 108_4 from FVC2002 : 0.50175
Computing similarity between 108_2.tif and 108_5 from FVC2002 : 0.48444
Computing similarity between 108_2.tif and 108_6 from FVC2002 : 0.58611
Computing similarity between 108_2.tif and 108_7 from FVC2002 : 0.58662
Computing similarity between 108_2.tif and 108_8 from FVC2002 : 0.64678
Computing similarity between 108_3.tif and 108_4 from FVC2002 : 0.70784
Computing similarity between 108_3.tif and 108_5 from FVC2002 : 0.4834
Computing similarity between 108_3.tif and 108_6 from FVC2002 : 0.64824
Computing similarity between 108_3.tif and 108_7 from FVC2002 : 0.6366

Computing similarity between 108_3.tif and 108_8 from FVC2002 : 0.6987
Computing similarity between 108_4.tif and 108_5 from FVC2002 : 0.45642
Computing similarity between 108_4.tif and 108_6 from FVC2002 : 0.57649
Computing similarity between 108_4.tif and 108_7 from FVC2002 : 0.58866
Computing similarity between 108_4.tif and 108_8 from FVC2002 : 0.60501
Computing similarity between 108_5.tif and 108_6 from FVC2002 : 0.45407
Computing similarity between 108_5.tif and 108_7 from FVC2002 : 0.52099
Computing similarity between 108_5.tif and 108_8 from FVC2002 : 0.53678
Computing similarity between 108_6.tif and 108_7 from FVC2002 : 0.66854
Computing similarity between 108_6.tif and 108_8 from FVC2002 : 0.7504
Computing similarity between 108_7.tif and 108_8 from FVC2002 : 0.76471
Computing similarity between 109_1.tif and 109_2 from FVC2002 : 0.69102
Computing similarity between 109_1.tif and 109_3 from FVC2002 : 0.76838
Computing similarity between 109_1.tif and 109_4 from FVC2002 : 0.67857
Computing similarity between 109_1.tif and 109_5 from FVC2002 : 0.73696
Computing similarity between 109_1.tif and 109_6 from FVC2002 : 0.37954
Computing similarity between 109_1.tif and 109_7 from FVC2002 : 0.59658
Computing similarity between 109_1.tif and 109_8 from FVC2002 : 0.79358
Computing similarity between 109_2.tif and 109_3 from FVC2002 : 0.61237
Computing similarity between 109_2.tif and 109_4 from FVC2002 : 0.80013
Computing similarity between 109_2.tif and 109_5 from FVC2002 : 0.75048
Computing similarity between 109_2.tif and 109_6 from FVC2002 : 0.31623
Computing similarity between 109_2.tif and 109_7 from FVC2002 : 0.50032
Computing similarity between 109_2.tif and 109_8 from FVC2002 : 0.59732
Computing similarity between 109_3.tif and 109_4 from FVC2002 : 0.70156
Computing similarity between 109_3.tif and 109_5 from FVC2002 : 0.72219
Computing similarity between 109_3.tif and 109_6 from FVC2002 : 0.48412
Computing similarity between 109_3.tif and 109_7 from FVC2002 : 0.59088
Computing similarity between 109_3.tif and 109_8 from FVC2002 : 0.80687
Computing similarity between 109_4.tif and 109_5 from FVC2002 : 0.84223
Computing similarity between 109_4.tif and 109_6 from FVC2002 : 0.37954
Computing similarity between 109_4.tif and 109_7 from FVC2002 : 0.4913
Computing similarity between 109_4.tif and 109_8 from FVC2002 : 0.72457
Computing similarity between 109_5.tif and 109_6 from FVC2002 : 0.44074
Computing similarity between 109_5.tif and 109_7 from FVC2002 : 0.55172
Computing similarity between 109_5.tif and 109_8 from FVC2002 : 0.71197
Computing similarity between 109_6.tif and 109_7 from FVC2002 : 0.40684
Computing similarity between 109_6.tif and 109_8 from FVC2002 : 0.33333
Computing similarity between 109_7.tif and 109_8 from FVC2002 : 0.57635
Computing similarity between 110_1.tif and 110_2 from FVC2002 : 0.75753
Computing similarity between 110_1.tif and 110_3 from FVC2002 : 0.87551
Computing similarity between 110_1.tif and 110_4 from FVC2002 : 0.43015
Computing similarity between 110_1.tif and 110_5 from FVC2002 : 0.30124
Computing similarity between 110_1.tif and 110_6 from FVC2002 : 0.55337
Computing similarity between 110_1.tif and 110_7 from FVC2002 : 0.33385
Computing similarity between 110_1.tif and 110_8 from FVC2002 : 0.38949
Computing similarity between 110_2.tif and 110_3 from FVC2002 : 0.68115
Computing similarity between 110_2.tif and 110_4 from FVC2002 : 0.41275
Computing similarity between 110_2.tif and 110_5 from FVC2002 : 0.33352
Computing similarity between 110_2.tif and 110_6 from FVC2002 : 0.53609
Computing similarity between 110_2.tif and 110_7 from FVC2002 : 0.36962
Computing similarity between 110_2.tif and 110_8 from FVC2002 : 0.40043
Computing similarity between 110_3.tif and 110_4 from FVC2002 : 0.56728
Computing similarity between 110_3.tif and 110_5 from FVC2002 : 0.56882
Computing similarity between 110_3.tif and 110_6 from FVC2002 : 0.49758
Computing similarity between 110_3.tif and 110_7 from FVC2002 : 0.50031
Computing similarity between 110_3.tif and 110_8 from FVC2002 : 0.4753
Computing similarity between 110_4.tif and 110_5 from FVC2002 : 0.68936
Computing similarity between 110_4.tif and 110_6 from FVC2002 : 0.37689
Computing similarity between 110_4.tif and 110_7 from FVC2002 : 0.60634
Computing similarity between 110_4.tif and 110_8 from FVC2002 : 0.66697
Computing similarity between 110_5.tif and 110_6 from FVC2002 : 0.31672
Computing similarity between 110_5.tif and 110_7 from FVC2002 : 0.76432
Computing similarity between 110_5.tif and 110_8 from FVC2002 : 0.63693
Computing similarity between 110_6.tif and 110_7 from FVC2002 : 0.36564
Computing similarity between 110_6.tif and 110_8 from FVC2002 : 0.32907
Computing similarity between 110_7.tif and 110_8 from FVC2002 : 0.70588

Table 7.5 below illustartes the full test log obtained during the execution of test case 2 in both CFAS and PFAS. Please refer to section 7.2.1.2 for the details of test case1. Each line in the test log shows a matching or comparison score (threshold *t*) obtained after

computing the similarity between two fingerprint images of the FVC2002-DB1_B database. The value of threshold *t* ranges between 0 and 1.

Table 7.5: Test log of Test case 2

CFAS log:

Computing similarity between 101_1.tif and 102_1 from FVC2002 : 0.19739
Computing similarity between 101_1.tif and 103_1 from FVC2002 : 0.18019
Computing similarity between 101_1.tif and 104_1 from FVC2002 : 0.24772
Computing similarity between 101_1.tif and 105_1 from FVC2002 : 0.19508
Computing similarity between 101_1.tif and 106_1 from FVC2002 : 0.29854
Computing similarity between 101_1.tif and 107_1 from FVC2002 : 0.30151
Computing similarity between 101_1.tif and 108_1 from FVC2002 : 0.28651
Computing similarity between 101_1.tif and 109_1 from FVC2002 : 0.24175
Computing similarity between 101_1.tif and 110_1 from FVC2002 : 0.2103
Computing similarity between 102_1.tif and 103_1 from FVC2002 : 0.31298
Computing similarity between 102_1.tif and 104_1 from FVC2002 : 0.31873
Computing similarity between 102_1.tif and 105_1 from FVC2002 : 0.40003
Computing similarity between 102_1.tif and 106_1 from FVC2002 : 0.3241
Computing similarity between 102_1.tif and 107_1 from FVC2002 : 0.3546
Computing similarity between 102_1.tif and 108_1 from FVC2002 : 0.28277
Computing similarity between 102_1.tif and 109_1 from FVC2002 : 0.29161
Computing similarity between 102_1.tif and 110_1 from FVC2002 : 0.27904
Computing similarity between 103_1.tif and 104_1 from FVC2002 : 0.21822
Computing similarity between 103_1.tif and 105_1 from FVC2002 : 0.28355
Computing similarity between 103_1.tif and 106_1 from FVC2002 : 0.3077
Computing similarity between 103_1.tif and 107_1 from FVC2002 : 0.29881
Computing similarity between 103_1.tif and 108_1 from FVC2002 : 0.2478
Computing similarity between 103_1.tif and 109_1 from FVC2002 : 0.25555
Computing similarity between 103_1.tif and 110_1 from FVC2002 : 0.33346
Computing similarity between 104_1.tif and 105_1 from FVC2002 : 0.25594
Computing similarity between 104_1.tif and 106_1 from FVC2002 : 0.36155
Computing similarity between 104_1.tif and 107_1 from FVC2002 : 0.38797
Computing similarity between 104_1.tif and 108_1 from FVC2002 : 0.29967
Computing similarity between 104_1.tif and 109_1 from FVC2002 : 0.31717
Computing similarity between 104_1.tif and 110_1 from FVC2002 : 0.33958
Computing similarity between 105_1.tif and 106_1 from FVC2002 : 0.32031
Computing similarity between 105_1.tif and 107_1 from FVC2002 : 0.29654
Computing similarity between 105_1.tif and 108_1 from FVC2002 : 0.18631
Computing similarity between 105_1.tif and 109_1 from FVC2002 : 0.20174
Computing similarity between 105_1.tif and 110_1 from FVC2002 : 0.32592
Computing similarity between 106_1.tif and 107_1 from FVC2002 : 0.3218
Computing similarity between 106_1.tif and 108_1 from FVC2002 : 0.34214
Computing similarity between 106_1.tif and 109_1 from FVC2002 : 0.29109
Computing similarity between 106_1.tif and 110_1 from FVC2002 : 0.32229
Computing similarity between 107_1.tif and 108_1 from FVC2002 : 0.32395
Computing similarity between 107_1.tif and 109_1 from FVC2002 : 0.30067
Computing similarity between 107_1.tif and 110_1 from FVC2002 : 0.3778
Computing similarity between 108_1.tif and 109_1 from FVC2002 : 0.32323
Computing similarity between 108_1.tif and 110_1 from FVC2002 : 0.44186
Computing similarity between 109_1.tif and 110_1 from FVC2002 : 0.37282

PFAS Log:

Computing similarity between 101_1.tif and 102_1 from FVC2002 : 0.22259
Computing similarity between 101_1.tif and 103_1 from FVC2002 : 0.21147
Computing similarity between 101_1.tif and 104_1 from FVC2002 : 0.26698
Computing similarity between 101_1.tif and 105_1 from FVC2002 : 0.22004
Computing similarity between 101_1.tif and 106_1 from FVC2002 : 0.34699
Computing similarity between 101_1.tif and 107_1 from FVC2002 : 0.21779
Computing similarity between 101_1.tif and 108_1 from FVC2002 : 0.30569
Computing similarity between 101_1.tif and 109_1 from FVC2002 : 0.27584
Computing similarity between 101_1.tif and 110_1 from FVC2002 : 0.23678
Computing similarity between 102_1.tif and 103_1 from FVC2002 : 0.36088
Computing similarity between 102_1.tif and 104_1 from FVC2002 : 0.29288
Computing similarity between 102_1.tif and 105_1 from FVC2002 : 0.36784
Computing similarity between 102_1.tif and 106_1 from FVC2002 : 0.31722
Computing similarity between 102_1.tif and 107_1 from FVC2002 : 0.34511
Computing similarity between 102_1.tif and 108_1 from FVC2002 : 0.27946
Computing similarity between 102_1.tif and 109_1 from FVC2002 : 0.23056
Computing similarity between 102_1.tif and 110_1 from FVC2002 : 0.3216
Computing similarity between 103_1.tif and 104_1 from FVC2002 : 0.23806
Computing similarity between 103_1.tif and 105_1 from FVC2002 : 0.30579
Computing similarity between 103_1.tif and 106_1 from FVC2002 : 0.32817

Computing similarity between 103_1.tif and 107_1 from FVC2002 : 0.29424
Computing similarity between 103_1.tif and 108_1 from FVC2002 : 0.2478
Computing similarity between 103_1.tif and 109_1 from FVC2002 : 0.25555
Computing similarity between 103_1.tif and 110_1 from FVC2002 : 0.35647
Computing similarity between 104_1.tif and 105_1 from FVC2002 : 0.21233
Computing similarity between 104_1.tif and 106_1 from FVC2002 : 0.33735
Computing similarity between 104_1.tif and 107_1 from FVC2002 : 0.40119
Computing similarity between 104_1.tif and 108_1 from FVC2002 : 0.2972
Computing similarity between 104_1.tif and 109_1 from FVC2002 : 0.33875
Computing similarity between 104_1.tif and 110_1 from FVC2002 : 0.31155
Computing similarity between 105_1.tif and 106_1 from FVC2002 : 0.29268
Computing similarity between 105_1.tif and 107_1 from FVC2002 : 0.31492
Computing similarity between 105_1.tif and 108_1 from FVC2002 : 0.18418
Computing similarity between 105_1.tif and 109_1 from FVC2002 : 0.19943
Computing similarity between 105_1.tif and 110_1 from FVC2002 : 0.29347
Computing similarity between 106_1.tif and 107_1 from FVC2002 : 0.3621
Computing similarity between 106_1.tif and 108_1 from FVC2002 : 0.35578
Computing similarity between 106_1.tif and 109_1 from FVC2002 : 0.28828
Computing similarity between 106_1.tif and 110_1 from FVC2002 : 0.31494
Computing similarity between 107_1.tif and 108_1 from FVC2002 : 0.319
Computing similarity between 107_1.tif and 109_1 from FVC2002 : 0.32898
Computing similarity between 107_1.tif and 110_1 from FVC2002 : 0.36711
Computing similarity between 108_1.tif and 109_1 from FVC2002 : 0.30014
Computing similarity between 108_1.tif and 110_1 from FVC2002 : 0.43601
Computing similarity between 109_1.tif and 110_1 from FVC2002 : 0.36788

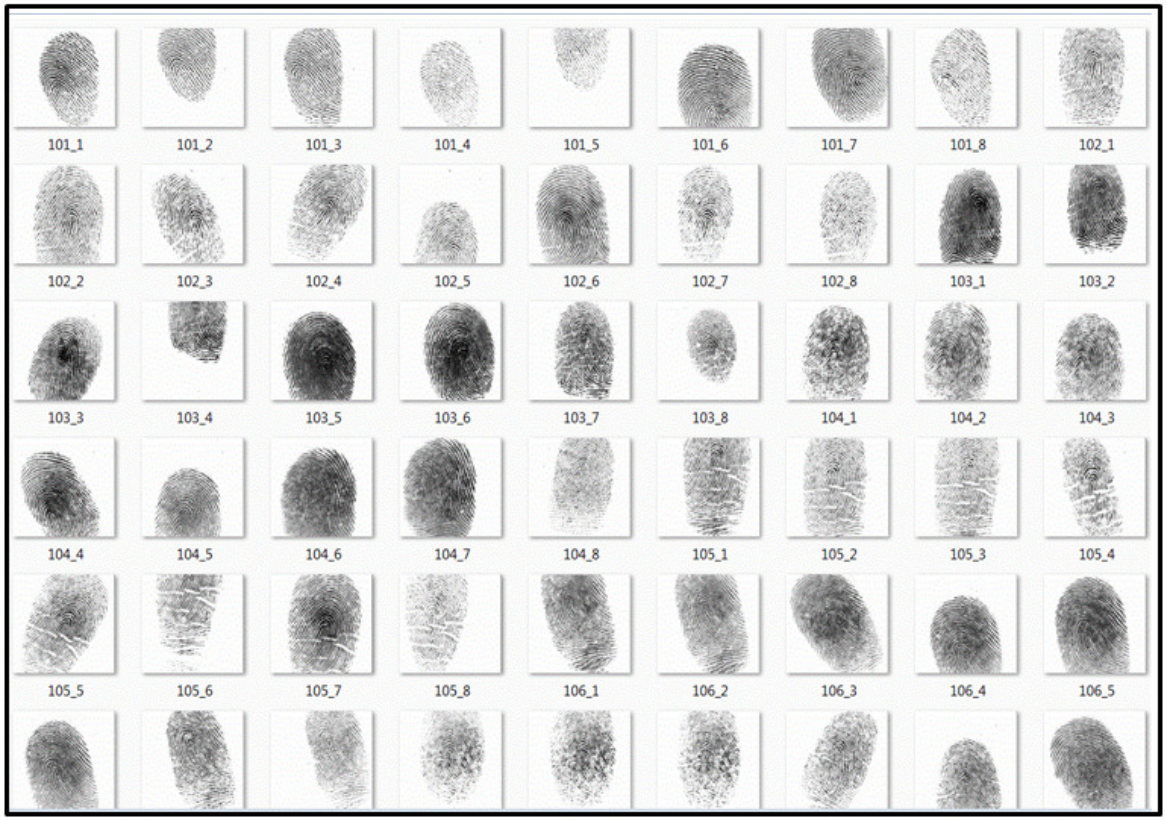
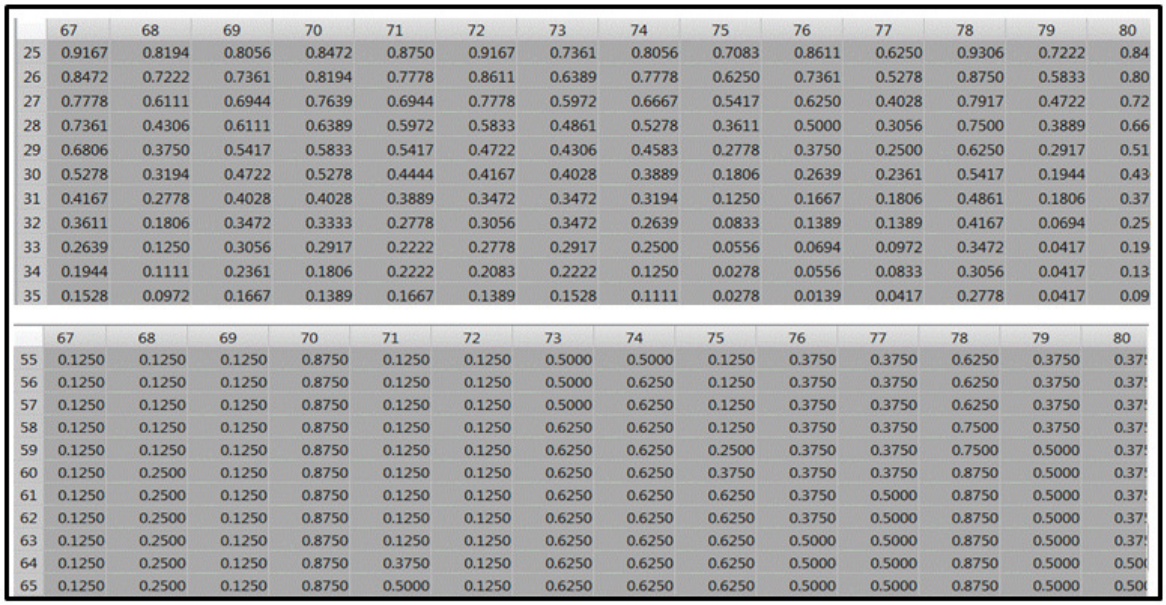


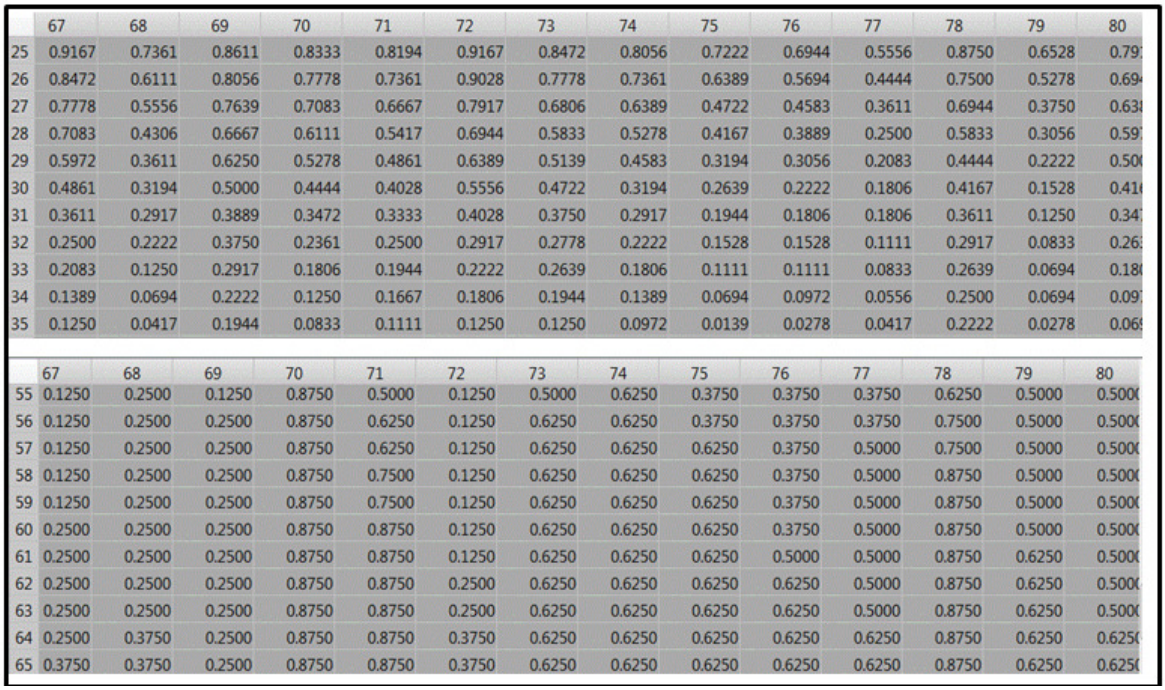
Figure 7.2: FVC2002-DB1_B



	67	68	69	70	71	72	73	74	75	76	77	78	79	80
25	0.9167	0.8194	0.8056	0.8472	0.8750	0.9167	0.7361	0.8056	0.7083	0.8611	0.6250	0.9306	0.7222	0.84
26	0.8472	0.7222	0.7361	0.8194	0.7778	0.8611	0.6389	0.7778	0.6250	0.7361	0.5278	0.8750	0.5833	0.80
27	0.7778	0.6111	0.6944	0.7639	0.6944	0.7778	0.5972	0.6667	0.5417	0.6250	0.4028	0.7917	0.4722	0.72
28	0.7361	0.4306	0.6111	0.6389	0.5972	0.5833	0.4861	0.5278	0.3611	0.5000	0.3056	0.7500	0.3889	0.66
29	0.6806	0.3750	0.5417	0.5833	0.5417	0.4722	0.4306	0.4583	0.2778	0.3750	0.2500	0.6250	0.2917	0.51
30	0.5278	0.3194	0.4722	0.5278	0.4444	0.4167	0.4028	0.3889	0.1806	0.2639	0.2361	0.5417	0.1944	0.43
31	0.4167	0.2778	0.4028	0.4028	0.3889	0.3472	0.3472	0.3194	0.1250	0.1667	0.1806	0.4861	0.1806	0.37
32	0.3611	0.1806	0.3472	0.3333	0.2778	0.3056	0.3472	0.2639	0.0833	0.1389	0.1389	0.4167	0.0694	0.25
33	0.2639	0.1250	0.3056	0.2917	0.2222	0.2778	0.2917	0.2500	0.0556	0.0694	0.0972	0.3472	0.0417	0.19
34	0.1944	0.1111	0.2361	0.1806	0.2222	0.2083	0.2222	0.1250	0.0278	0.0556	0.0833	0.3056	0.0417	0.13
35	0.1528	0.0972	0.1667	0.1389	0.1667	0.1389	0.1528	0.1111	0.0278	0.0139	0.0417	0.2778	0.0417	0.09

	67	68	69	70	71	72	73	74	75	76	77	78	79	80
55	0.1250	0.1250	0.1250	0.8750	0.1250	0.1250	0.5000	0.5000	0.1250	0.3750	0.3750	0.6250	0.3750	0.37
56	0.1250	0.1250	0.1250	0.8750	0.1250	0.1250	0.5000	0.6250	0.1250	0.3750	0.3750	0.6250	0.3750	0.37
57	0.1250	0.1250	0.1250	0.8750	0.1250	0.1250	0.5000	0.6250	0.1250	0.3750	0.3750	0.6250	0.3750	0.37
58	0.1250	0.1250	0.1250	0.8750	0.1250	0.1250	0.6250	0.6250	0.1250	0.3750	0.3750	0.7500	0.3750	0.37
59	0.1250	0.1250	0.1250	0.8750	0.1250	0.1250	0.6250	0.6250	0.2500	0.3750	0.3750	0.7500	0.5000	0.37
60	0.1250	0.2500	0.1250	0.8750	0.1250	0.1250	0.6250	0.6250	0.3750	0.3750	0.3750	0.8750	0.5000	0.37
61	0.1250	0.2500	0.1250	0.8750	0.1250	0.1250	0.6250	0.6250	0.6250	0.3750	0.5000	0.8750	0.5000	0.37
62	0.1250	0.2500	0.1250	0.8750	0.1250	0.1250	0.6250	0.6250	0.6250	0.3750	0.5000	0.8750	0.5000	0.37
63	0.1250	0.2500	0.1250	0.8750	0.1250	0.1250	0.6250	0.6250	0.6250	0.5000	0.5000	0.8750	0.5000	0.37
64	0.1250	0.2500	0.1250	0.8750	0.3750	0.1250	0.6250	0.6250	0.6250	0.5000	0.5000	0.8750	0.5000	0.50
65	0.1250	0.2500	0.1250	0.8750	0.5000	0.1250	0.6250	0.6250	0.6250	0.5000	0.5000	0.8750	0.5000	0.50

Figure 7.9: Subset of the FAR/FRR tables generated in the CFAS



	67	68	69	70	71	72	73	74	75	76	77	78	79	80
25	0.9167	0.7361	0.8611	0.8333	0.8194	0.9167	0.8472	0.8056	0.7222	0.6944	0.5556	0.8750	0.6528	0.79
26	0.8472	0.6111	0.8056	0.7778	0.7361	0.9028	0.7778	0.7361	0.6389	0.5694	0.4444	0.7500	0.5278	0.69
27	0.7778	0.5556	0.7639	0.7083	0.6667	0.7917	0.6806	0.6389	0.4722	0.4583	0.3611	0.6944	0.3750	0.63
28	0.7083	0.4306	0.6667	0.6111	0.5417	0.6944	0.5833	0.5278	0.4167	0.3889	0.2500	0.5833	0.3056	0.59
29	0.5972	0.3611	0.6250	0.5278	0.4861	0.6389	0.5139	0.4583	0.3194	0.3056	0.2083	0.4444	0.2222	0.50
30	0.4861	0.3194	0.5000	0.4444	0.4028	0.5556	0.4722	0.3194	0.2639	0.2222	0.1806	0.4167	0.1528	0.41
31	0.3611	0.2917	0.3889	0.3472	0.3333	0.4028	0.3750	0.2917	0.1944	0.1806	0.1806	0.3611	0.1250	0.34
32	0.2500	0.2222	0.3750	0.2361	0.2500	0.2917	0.2778	0.2222	0.1528	0.1528	0.1111	0.2917	0.0833	0.26
33	0.2083	0.1250	0.2917	0.1806	0.1944	0.2222	0.2639	0.1806	0.1111	0.1111	0.0833	0.2639	0.0694	0.18
34	0.1389	0.0694	0.2222	0.1250	0.1667	0.1806	0.1944	0.1389	0.0694	0.0972	0.0556	0.2500	0.0694	0.09
35	0.1250	0.0417	0.1944	0.0833	0.1111	0.1250	0.1250	0.0972	0.0139	0.0278	0.0417	0.2222	0.0278	0.06

	67	68	69	70	71	72	73	74	75	76	77	78	79	80
55	0.1250	0.2500	0.1250	0.8750	0.5000	0.1250	0.5000	0.6250	0.3750	0.3750	0.3750	0.6250	0.5000	0.500
56	0.1250	0.2500	0.2500	0.8750	0.6250	0.1250	0.6250	0.6250	0.3750	0.3750	0.3750	0.7500	0.5000	0.500
57	0.1250	0.2500	0.2500	0.8750	0.6250	0.1250	0.6250	0.6250	0.6250	0.3750	0.5000	0.7500	0.5000	0.500
58	0.1250	0.2500	0.2500	0.8750	0.7500	0.1250	0.6250	0.6250	0.6250	0.3750	0.5000	0.8750	0.5000	0.500
59	0.1250	0.2500	0.2500	0.8750	0.7500	0.1250	0.6250	0.6250	0.6250	0.3750	0.5000	0.8750	0.5000	0.500
60	0.2500	0.2500	0.2500	0.8750	0.8750	0.1250	0.6250	0.6250	0.6250	0.3750	0.5000	0.8750	0.5000	0.500
61	0.2500	0.2500	0.2500	0.8750	0.8750	0.1250	0.6250	0.6250	0.6250	0.5000	0.5000	0.8750	0.6250	0.500
62	0.2500	0.2500	0.2500	0.8750	0.8750	0.2500	0.6250	0.6250	0.6250	0.6250	0.5000	0.8750	0.6250	0.500
63	0.2500	0.2500	0.2500	0.8750	0.8750	0.2500	0.6250	0.6250	0.6250	0.6250	0.5000	0.8750	0.6250	0.500
64	0.2500	0.3750	0.2500	0.8750	0.8750	0.3750	0.6250	0.6250	0.6250	0.6250	0.6250	0.8750	0.6250	0.625
65	0.3750	0.3750	0.2500	0.8750	0.8750	0.3750	0.6250	0.6250	0.6250	0.6250	0.6250	0.8750	0.6250	0.625

Figure 7.10: Subset of the FAR/FRR tables generated in the PFAS