Enhancing the Security of Centralised and Distributed Payments

Submitted by

Danushka Jayasinghe

for the degree of Doctor of Philosophy

of the

Royal Holloway, University of London

2017

Declaration

I, Danushka Jayasinghe, hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Signed . . . . (Danushka Jayasinghe)
Date:

Publications

The research carried out has led to the publication and presentation of a number of academic papers in international conferences.

1. D. Jayasinghe, K. Markantonakis and K. Mayes, “Optimistic Fair-Exchange with Anonymity for Bitcoin Users,” In *The 11th International Conference on e-Business Engineering (ICEBE)*, Guangzhou, China, pp.44-51, IEEE, 2014.

2. D. Jayasinghe, R. N. Akram, K. Markantonakis, K. Rantos and K. Mayes, “Enhancing EMV Online PIN Verification,” In *The 14th International Conference on Trust, Security and Privacy in Computing and Communications (Trustcom/ BigDataSE/ISPA)*, pp.808-817, Helsinki, Finland, IEEE, 2015.

3. D. Jayasinghe, K. Markantonakis, I. Gurulian, R. N. Akram and K. Mayes, “Extending EMV Tokenised Payments to Offline-Environments,” In *The 15th International Conference on Trust, Security and Privacy in Computing and Communications (Trustcom/BigDataSE/ISPA)*, pp.443-450, Tianjin, China, IEEE, 2016.

4. D. Jayasinghe, K. Markantonakis, R. N. Akram and K. Mayes, “Enhancing EMV Tokenisation with Dynamic Transaction Tokens,” In *12th International Workshop: Radio Frequency Identification and IoT Security (RFIDSec)*, pp.107-122, Hong Kong, Springer, 2016.

5. D. Jayasinghe, S. Cobourne, K. Markantonakis, R. N. Akram and K. Mayes, “Philanthropy On The Blockchain,” In *The 11th WISTP International Conference on Information Security Theory and Practice (WISTP)*, Heraklion, Greece, Springer, 2017.

Abstract

In the last few decades, the way humans engage in payment transactions and the tools they use to transact with each other have evolved dramatically. Advancement in cryptography, information security, computer networking, distributed computing, etc. provides the required tools for modern day payment solution providers to design payment technologies that can be used to carry out convenient payment transactions. Yet, the financial loss associated with financial fraud, payment related attacks and data breaches that directly affect the financial institutions, merchants and consumers is significant. Because of this, an important development in the payment evolution is the consideration towards security of payment transactions.

The main focus of this thesis is to enhance the security of both EMV (Europay MasterCard Visa) based centralised payments and Bitcoin/blockchain based distributed payments while showing more emphasis on new and emerging payment technologies such as: mobile payments, tokenisation and distributed ledger technology.

EMV is a standard that provides interoperability to Chip & PIN, Contactless and Tokenised payment transactions in a global scale. The thesis, investigates the current EMV payment architectures to identify potential weaknesses that pose a threat to the security of payment transactions. In our research, we were able to identify five main issues related to EMV Online PIN Verification in two deployment methods and three main issues related to EMV Tokenisation that raise security concerns. We discuss potential attack scenarios, and propose solutions that address the identified issues and enhance the security of payment transactions. The proposed solutions are subject to mechanical formal analysis and practical implementation was carried out to obtain performance measurements.

The thesis, also investigates payments in distributed payment systems such as Bitcoin and blockchains. We identify issues such as fair-exchange related to distributed payments and propose solutions to improve security and anonymity. Furthermore, we explore how blockchain technology can be leveraged to enhance the security in other payment transactions such as: donation payments, humanitarian aid and SMS-based mobile payments. Finally, the thesis provides conclusion of this research and suggesting future research directions.

Acknowledgement

The completion of this thesis would not have been possible without the support of a number of wonderful people. First and foremost, I would like to express my deepest gratitude to my supervisor Prof. Konstantinos Markantonakis for his supervision and guidance throughout my PhD. I am grateful for his approachability, invaluable reviews and feedback. I would also like to thank my advisor, Prof. Keith Mayes for his support during the PhD.

I would like to thank Dr. Raja Naeem Akram for his insightful reviews and comments on the thesis. I am also grateful to Sheila Cobourne for her reviews and support throughout my PhD.

Special thanks to all my colleagues at the Smart Card & IoT Security Centre: Benoit Ducray, Iakovos Gurulian, Assad Umar, Lazaros Kyrillidis, Sara Abu-Ghazalah, Hafizah Mansor, Mehari Msgna, Rashedul Hassan, Robert Lee and Carlton Shepherd for the many interesting discussions.

A very special thank goes to my beloved wife Madushani for her amazing support, love and care during my PhD. Your encouragement was a key ingredient for completing this milestone. I would also like to thank my parents and my two sisters for their love, support and encouragement during this time.

Even though there isn't enough space here to mention all your names, I would also like to thank my friends, relatives, in-laws and others who have provided me help and support in any shape or form during this remarkable journey.

Finally, I would also like to thank the UK Cards Association (subset of UK Finance since July 2017) and the ISG - Smart Card & IoT Security Centre for financially sponsoring my PhD. The invaluable discussions, recommendations and feedback provided by David Baker and Bill Reding from the UK Cards Association throughout this work are much appreciated.

Contents

List of Figures

List of Tables

List of Abbreviations

ATM	Automated Teller Machine.
AES	Advanced Encryption Standard.
API	Application Programming Interfaces.
ARC	Authorisation Response Code.
ARQC	Authorisation Request Cryptogram.
ARPC	Authorisation Response Cryptogram.
BTC	Bitcoin Currency.
BPS	Bitcoin Payment Server.
CA	Certification Authority.
CNP	Card Not Present.
CDA	Combined Data Authentication
CIB	Card Issuing Bank.
CVM	Cardholder Verification Method.
CFB	Cipher Feedback Mode.
CUN	Card Unpredictable Number.
CVR	Cardholder Verification Result.
CSP	Communicating Sequential Processes.
CTPOS	Payment Terminal at Point of Sale.
DDA	Dynamic Data Authentication.
DTD	Dynamic Token Data.
DTT	Dynamic Transaction Token.
DDoS	Distributed Denial of Service.
EMV	Europay MasterCard Visa.
ECDSA	Elliptic Curve Digital Signature Algorithm.
FFA	Financial Fraud Action UK.
FDR	Failures-Divergence Refinement.
GUI	Graphical User Interface.
GCM	Galois/Counter Mode.
HCE	Host Card Emulation.
HOTP	Hash-based One Time Password.
IV	Initialisation Vector.
IFD	Interface Device.

MAC	Message Authentication Code.
MNO	Mobile Network Operator.
NFC	Near Field Communication.
NPO	Non Profit Organisation.
OPV	Online-PIN Verification.
OTT	Offline Transaction Token.
OTA	Over-The-Air.
OTP	One Time Password .
PB	OPV PIN Block.
PIN	Personal Identification Number.
PAN	Primary Account Number.
PoS	Point-of-Sale.
PTO	Payment Terminal Operator.
PRB	OPV PIN Result Block.
P2SH	Pay To Script Hash.
PRNG	Pseudo Random Number Generator.
PKCS	Public Key Cryptography Standards.
PDOL	Processing Options Data Object List.
PSI-DSS	Payment Card Industry-Data Security Standard.
RSA	Rivest, Shamir and Adleman.
RSK	Rootstock.
SO	Scheme Operator.
SE	Secure Element.
SMS	Short Message Service.
SHA	Secure Hashing Algorithm.
SDA	Static Data Authentication.
SBTC	Smart Bitcoin Currency.
SPDL	Security Protocol Description Language.
TC	Transaction Certificate.
TTP	Trusted Third Party.
TAR	Token Authorisation Request.
TSP	Token Service Provider.
TVR	Terminal Verification Result.
UN	Unpredictable Number.

Chapter 1

Introduction

Contents

The innovation that comes out from payment technology not only helps improve the economy but also enhances the quality and security of consumer-merchant transactions. This chapter, sets the scene by explaining the motivation and challenges behind this thesis. Following this, the main contributions on improving the security of payment transactions are briefly discussed. Finally, the chapter concludes by outlining the structure of the thesis for the remaining chapters.

1.1 Motivation and Challenges

Modern technological advancements have dramatically boosted how technological knowledge is shared globally within a very short period of time than it used to be. This has led to introduction of systems that provide global interoperability. One industry that has harnessed this opportunity is the payment industry.

In the last few decades, how humans engage in payment transactions and the tools they use to transact have evolved dramatically. One significant development in the payment evolution is the consideration towards security of payment transactions. Advancement in cryptography, information security, computer networking, distributed computing, etc. provides the required tools for modern day payment solution providers to design payment technologies that can be used to carry out convenient payment transactions. Most of the widely used centralised payment systems are standardised and regulated with stringent security controls. Yet, the financial loss associated with financial fraud, payment related attacks and data breaches that directly affect the financial institutions, merchants and consumers is significant. This applies to both centralised and distributed payment systems.

Furthermore, planned/unplanned changes to payment architectures such as: introduction of a large number of intermediaries to payment channels, technological advancements available to adversaries, identified vulnerabilities, etc. pose a new threat to the security of payment technologies that were considered secure in the past. Taking these concerns into consideration, we raise the following research questions in this thesis.

Are modern day payment transactions secure and provide the expected security guarantees? Are there still shortcomings and weaknesses in widely used payment systems that we consider secure? Finally, is there still space for improvement to enhance the security of both centralised and distributed payment transactions?

Investigating and finding answers to the aforementioned research questions will be the main aim of the thesis. Even though, it is extremely difficult to provide absolute security, every effort must be taken to address the current threat presented against the security of a payment technology at a given time. Addressing such issues will enhance the security of payment transactions and prevent unnecessary loss associated with financial fraud.

The research carried out will explore current and emerging payment technologies such as: EMV (Europay Mastercard Visa) Chip & PIN, Contactless Card/Mobile Payments, Tokenisation, Digital Currencies and Blockchain. In particular, current payment technologies such as: Chip & PIN and Contactless Card/Mobile Payments were given more emphasis then other payment technologies, because of global usage, industrial adoption and the significance of the number of stakeholders that would benefit from our proposed improvements, etc. Emerging payment technologies such as: EMV Tokenisation, Digital Currencies and Blockchain were given more emphasis because of innovative technologies used by these payment systems, increased adoption in the payment industry at the time of writing and the potential of being a disruptive technology for providing secure payment transaction solutions, etc. In this thesis, a number of payment technologies are investigated to identify weaknesses/vulnerabilities that could potentially lead to security concerns and propose solutions that address the identified issues and further enhance the security of payment transactions.

The proposed solutions are expected to strengthen the security of associate payment protocols, in order to enhance the security of payment transactions.

1.2 Aim and Objectives

The main aim of the thesis is to enhance the security of centralised and distributed payment transactions. This include finding answers to the research questions that were raised in the previous Section 1.1 by investigating security aspects of both centralised and distributed payment transactions and proposing improvements that address identified security concerns and limitations.

In order to achieve the main aim of the thesis, the following three main objectives of the thesis are identified:

Objective-1: Investigate payment transactions in both centralised and distributed payments while showing more emphasis on new/emerging payment technologies.

Objective-2 Identify potential weaknesses and concerns in payment technologies that pose a threat to the security of payment transactions.

Objective-3 Propose improvements that address these identified weaknesses and concerns to enhance the security of payment transactions.

1.3 Contributions

In this section, we briefly discuss our main contributions of the thesis.

In this thesis, we propose improvements to enhance the security of both EMV based Centralised Payments and Bitcoin/Blockchain based Distributed Payments. In particular, we show more emphasis on current and emerging payment technologies. We first investigate a number of payment technologies, their underlying payment architectures and operating environments. By doing so, the current mechanisms that provide security guarantees of each payment system was established with the aim to understand what aspects can potentially be enhanced.

The thesis provides six main contributions in subsequent discussions. The contributions are divided into two main categories: Improvements for EMV based Centralised Payments and Improvements for Bitcoin/Blockchain based Distributed Payments. The contributions of the thesis are presented in Part II to Part III of this thesis.

Under the Improvements for EMV based Centralised Payments in Part II, we first investigated the EMV Online PIN Verification (OPV) process in the current payment architecture. In the first OPV deployment method that we investigate, the OPV process is carried out separately to the online transaction authorisation. We call this the Segmented Authorisation. We were able to identify a number of potential attack scenarios that pose a threat to the security of OPV process in the Segmented Authorisation method. Addressing the identified security concerns, our first contribution is the proposed protocol and improvements that enhance the security of OPV process.

Following this, we extended our work on enhancing EMV OPV by investigating another method that OPV process can be deployed in the current payment architecture. We call this second method the Unified Authorisation method. We then identify potential issues that can be used by an adversary to compromise the OPV process in the Unified Authorisation method. As our second contribution, we propose improvements and a protocol that addressed these issues to enhance the security of OPV in the Unified Authorisation method.

Next, we focused our attention on another aspect of the EMV payment architecture, which is the Transaction Authorisation. In our discussion, we explained about an inherent weakness in the current payment architecture which is the Primary Account Number (PAN) Compromise that has lead to significant financial loss for the financial institutions, merchants and the consumers. EMV Tokenisation is increasingly being adopted by the payments industry as a solution to PAN compromise. The background research and introduction to Tokenisation was given in Section 2.1.4 of the thesis. In our investigation of the current tokenisation architecture, however, we identified that the lack of support for making offline tokenised payments as a significant bottleneck in tokenised payment transactions. We then explained benefits of having the capability of making and accepting tokenised payment in an offline enviornment for both the merchants and the consumers. As our third contribution, we proposed a tokenisation based payment protocol which provides capability of making tokenised payments in offline environments.

Following on this, we extended our work on enhancing the security of tokenised payments by investigating the current tokenisation payment architecture. We were able to identify five potential attack scenarios that raise security concerns on tokenised payments, especially when a static-token is presented to the terminal during an EMV transaction. The security concerns include: over charging for a payment, capturing token related data, capturing the unpredictable number generated by the terminal, replay attack on authorisation response sent by the issuing bank and replay attack on the authorisation response even when the payment related data has been authorised offline before reaching the issuing bank. For our fourth contribution, we proposed a contactless mobile payment protocol that provides: mutual-authentication between the terminal and the mobile, security against the identified attack scenarios by using a proposed Dynamic Transaction Token (DTT) that is unique to a particular transaction and end-to-end encryption between the terminal and the payment authorisation entity as well as the terminal and the mobile. The improvements we proposed enhance the security of tokenised payment transactions and provide control for the authorisation entity to detect, prevent fraud and make informed payment authorisation decisions.

The proposed protocols under this category are subject to mechanical formal analysis for their security, where no feasible attacks were identified. Furthermore, a number of proposed solutions were implemented to obtain performance measurements.

Under the Improvements for Bitcoin/Blockchain based Distributed Payments in Part III of this thesis, we first explored the e-commerce market in an attempt to understand current trends and increased use of alternative payment methods such as digital cash that provide anonymity as an additional property. We then explained why it is difficult to guarantee fair-exchange in an e-commerce payment transaction compared to a traditional Point-of-Sale (PoS) payment transaction.

In our research, we identified a potential concern in anonymous payments such as Bitcoin when it comes to guaranteeing fair-exchange between a merchant and a consumer in an e-commerce transaction. The background research on anonymous payment protocols, fair-exchange payment protocols, Bitcoin and blockchain technology was carried out in Section 2.3 of the thesis. For our fifth contribution, we proposed a payment protocol that guarantees true fair-exchange during an e-commerce transaction when Bitcoin is used as an anonymous payment method. We then analyse the protocol for its security and anonymity requirements. Following this we carried out a discussion about Bitcoin transaction link-ability and explained how additional anonymity guarantees can be established. Finally, the proposed protocol was extended to support other cryptocurrencies such as Zerocoin/Zerocash that provide improved security and anonymity.

Next, we focused our attention onto other payment transaction scenarios that can leverage the blockchain technology. We identified the philanthropic sector as an industry that can gain benefits by using blockchains. In particular, donation payments associated with foreign or humanitarian aid activities. In our investigation to the current philanthropic models we were able to identify issues such as: donation transparency, transactional costs, speed of getting the donations to the beneficiaries, provisioning the received donation to beneficiaries in disconnected environments, etc. as main challenges faced by charities. As our sixth and final contribution, we proposed a novel philanthropic model that address these challenges by leveraging the Bitcoin blockchain for donating foreign aid for humanitarian causes. We then proposed a SMS based mobile payment solution to provision donations and to be used by beneficiaries in disconnected environments for their day-to-day payments. The solution is finally analysed for its security requirements.

Our work in this thesis identifies issues and weaknesses in payment systems used in real world that raise concerns regarding the security of centralised and distributed payment transactions. To address these issues, we carried out a background investigation on each payment technology and their underlying payment architectures. Addressing the identified security concerns, we proposed a number of solutions that improve the security of both centralised and distributed payments. Enhancing the security of payment transactions is primarily the main contribution of the thesis. However, we also consider improving privacy related to distributed payments. More specifically, anonymity which is a property of most cryptocurrencies, is a privacy element that we consider under distributed payments. In our proposed solutions, we discuss how anonymity is guaranteed or enhanced while improving the security of these payment transactions. Furthermore, the work carried out in this thesis has resulted in six academic submissions in international conferences and journals.

1.4 Thesis Outline

The rest of the thesis is organised as follows.

The work carried out in Part I of the thesis, presents background work related to both centralised and distributed payments. The beginning of Chapter 2 presents the background related to EMV based Centralised Payments. The discussion starts by giving an introduction to EMV and its practical use in the payments industry. Then a number of payment technologies such as: Chip & PIN, Contactless Card/Mobile Payments and Tokenisation are introduced. Following this, an investigation is carried out on each technology to identify their underlying architectures and the operating environments. Following this, Chapter 2 presents the background related to Bitcoin/Blockchain based

Distributed Payments. First, an introduction to Bitcoin, its architecture and security features is given. Afterwards, a discussion on the blockchain technology and key properties/aspects inherent with the technology is carried out. Following this, a number of proposals that leverage the blockchain technology to provide security related solutions are discussed. Chapter 2 then presents background related to Anonymous and Fair-exchange Payment Protocols. Finally, the chapter carries out a discussion on formal analysis of cryptographic protocols.

The work carried out in Part II of the thesis, present the proposed improvements for EMV based Centralised Payments. In Chapter 3, we present our first contribution of the thesis under centralised payments. In this chapter, the current EMV OPV process is discussed and potential issues related to the security of OPV is identified. We then propose solutions that addresses these issues and improve the security of OPV. The proposed protocol is mechanically analysed and implemented to obtain performance measurements. The research carried out in Chapter 3 has led to the publication of academic paper number two under the title "*Enhancing EMV Online PIN Verification*" in the list of publications.

In Chapter 4, we extend the work in our first contribution presented in Chapter 3 by introducing a second method that OPV can be deployed. Afterwards, we identify potential security issues in this deployment method and propose solutions that improve the security of OPV. Finally we analyse the proposed solution. Based on the research carried out in Chapter 4, paper number six under the title "*Enhancing EMV Online PIN Verification For A Second Deployment Method*" in the list of publications is under review at an international journal.

In Chapter 5, we discuss an inherent weakness in the current EMV architecture and introduce EMV Tokenisation which has been adopted as a solution. We then investigate the EMV tokenisation architecture and identify a potential drawback related to offline payments. Addressing this issue, we propose a tokenisation based payment protocol that can be used in offline environments. The proposed protocol is then subject to mechanical formal analysis and implemented to obtain performance measurements. The research carried out in Chapter 5 has led to the publication of academic paper number three under the title "*Extending EMV Tokenised Payments to Offline-Environments*" in the list of publications.

Chapter 6, investigates the security of tokenised payments. We identify a number of potential attack scenarios that pose a threat to tokenised based payment transactions. After this, we propose a contactless mobile payment protocol that is based on tokenisation to address the potential issues and improve the security of tokenised payment transactions. Finally, the proposed protocol is subject to mechanical formal analysis.

The published academic paper number four under the title "*Enhancing EMV Tokenisation with Dynamic Transaction Tokens*" in the list of publications was based on the research carried out in Chapter 6.

The work carried out in Part III of the thesis, present the proposed improvements for Bitcoin/Blockchain based Distributed Payments. More specifically in Chapter 7, we present our fifth contribution of the thesis. The work carried out in this Chapter 7, first discuss the e-commerce environment and introduce distributed payments. We then investigate anonymous payment protocols and fair-exchange payment protocols. Afterwards, we identify a potential issue when it comes to guaranteeing fairness in anonymous payments such as Bitcoin. We then propose a protocol that guarantees true fair exchange for Bitcoin payments in e-commerce. Finally, we analyse the protocol and extend it to support other cryptocurrencies such as Zerocoin/Zerocash. The research carried out in Chapter 7 has led to the publication of academic paper number one under the title "*Optimistic Fair-Exchange with Anonymity for Bitcoin Users*" in the list of publications.

Chapter 8, extends the work on Bitcoin/blockchain payment and explore other payment scenarios that can be enhanced by leveraging the blockchain technology. In particular, we focus our attention on donation payment involved with foreign/ humanitarian aid activities. We first identify issues and challenges faced by charities and then discuss the advantages of blockchain solutions. Following this, a novel philanthropic model that leverages the Bitcoin blockckchain was proposed. Furthermore, we propose a SMS based mobile payment solution to provision donations and to be used by beneficiaries in disconnected environments. The solution is finally analysed for its security requirements. The research carried out in Chapter 8 has led to the publication of academic paper number five under the title "*Philanthropy On The Blockchain*" in the list of publications. Finally, Chapter 9, provides concluding remarks of the thesis and outlines future research directions.

Part I

Background

Chapter 2

Payment Systems and Formal Analysis Tools

Contents

In this chapter, we present the background work related to both EMV based centralised payments and Bitcoin/blockchain based distributed payments. We explore the underlying payment architectures, operating environments and security features on a number of current and emerging payment technologies. Following this, we carry out background research related to anonymous and fair-exchange payment protocols. Finally, we discuss formal analysis of cryptographic protocols and introduce the chosen tools that will be used to formally analyse our proposed protocols in subsequent chapters.

2.1 EMV based Centralised Payments

Centralised payment systems are payment schemes that are controlled, owned, managed or regulated by any financial service, bank or a government. A good example of this is bank issued credit and debit card based payments. In the recent years there have been a lot of academic research carried out in this particular area [146, 88, 89, 59, 60, 70] as well as real world industrial solutions that has already been rolled out [13, 14, 15, 16, 24, 20]. A significant aspect is that due to the wide use of mobile phones, the traditional payment card based payments are now been gradually migrated to mobile phones. Compared to payment cards, mobile phones have added communication methods to run innovative payment applications but this has also given new attack vectors for perpetrators.

2.1.1 EMV

EMV (Europay MasterCard Visa) is a globally accepted standard, initially introduced for Chip & PIN payment transactions [14, 15] and contactless transactions [24].

In smart card-based payment systems (credit and debit) [159], security sensitive data related to both the payment system and the cardholder is securely stored on the tamper-resistant chip in the smart card [146, 133]. The Personal Identification Number (PIN) is used to establish an association between the smart card and its authorised cardholder during a chip & PIN transaction. In the EMV payment scheme, the PIN issued by a Card Issuing Bank (CIB) for a particular payment card is considered to be known only by the authorised cardholder, the issued payment card and the CIB or the payment authorisation entity.

In some scenarios the CIB will instruct the Scheme Operator (SO) which is a trusted entity that manages the payment scheme (e.g. MasterCard, Visa or Amex) to authorise EMV transactions on CIB's behalf. This process is generally known as the stand-in-process. In such situations, the PIN details are also shared with the SO. Furthermore, the authorised cardholder may share their PIN with other users such as family members. It must be noted that, in such scenarios, we consider that a payment transaction is initiated with the cardholder's consent.

During a EMV chip & PIN based payment transaction, the cardholder first inserts the payment card into the Terminal. The payment terminal discussed throughout the chapter is the payment acceptance device that is used to accept card-based payment transactions. Note that this is not an Automated Teller Machine (ATM). We refer to the terminal as CTPOS in this chapter.

Following this, the card and the terminal communicate with each other to engage in an EMV transaction. In EMV both the payment card and the terminal are set with liability limits such as maximum payment amount, risk measures and predefined parameters. The card and the terminal establish common parameters for the transaction based on their risk assurance levels. If this established common parameter require cardholder authentication, then the terminal requests the consumer to enter his/her PIN.

Depending on the PIN authorisation environment the entered PIN is verified either by the payment card or by the authorisation entity (i.e. scheme operator or CIB). Depending on the PIN verification result, the cardholder is either authorised to proceed with the transaction or not authorised. The PIN can be related to as an alternative to a cardholder's signature. The knowledge of the PIN entered by the cardholder during a transaction provides some assurance that the genuine cardholder is initiating the payment. The combination of using a smart card and its associated PIN can significantly reduce payment card fraud [182, 123].

EMV supports both offline and online PIN verification methods. In offline PIN verification the cardholder PIN is sent to the card either in plaintext or enciphered format to be verified. In contrast to this, during an Online-PIN Verification (OPV) the PIN needs to be sent to the authorisation entity (e.g. scheme operator or CIB) for verification. This process is explained in detail in the next section.

2.1.2 EMV Online PIN Verification

In this section we expand our discussion on the EMV OPV process. We first carried out a background research to understand how the OPV process works in the current EMV payment architecture. For this, resources such as: the EMV specifications and past literature was referred to.

We understood that the EMV specifications [13, 14, 15, 16] do not specify information related to the OPV process that may be supported by CTPOS devices. Similarly, to the best of our knowledge, there are no publicly available documentation that specifically detail the OPV process between the CTPOS and the authorising entity. However, there are some documentation such as [49, 131, 65] that explains how OPV as a Cardholder Verification Method is carried out during an Automated Teller Machine (ATM) transaction. We outline the OPV process carried out in the current architecture by considering what we have understood about the OPV process associated with ATM transactions and referring to resources such as [195].

In the OPV process, there may be a number of intermediary entities in the payment communication channel between the CTPOS and the authorisation entity. These intermediaries could be: the acquirer's subcontractors that manages the CTPOS devices, third party Payment Terminal Operators (PTO) and other nodes engaged in the key-translation mechanism (where messages are enciphered and deciphered from node to node).

The current EMV architecture has placed an indelible trust assumption on the intermediaries involved. Most of these intermediaries are bound by contracts with either the acquirer or SO/CIB, yet it is questionable whether this is sufficient to let intermediaries handle sensitive data related to the cards and cardholders. The EMV specification [16] states that "When the applicable Cardholder Verification Method (CVM) is online PIN, the Interface Device (IFD) shall not issue a Verify command. Instead the PIN pad shall encipher the PIN upon entry for transmission in the authorisation or financial transaction request".

Even though this indicates that the PIN is forwarded in encrypted format from the CTPOS onwards, it does not specifically detail any information related to the PIN encipherment for the OPV process. Neither does it mention whether the CTPOS encrypts the PIN with a cryptographic key only shared with the authorisation entity or with the next point of contact that engages in key-translation on the communication path.

The CTPOS devices are deployed by the PTO. The PTOs that the merchants use can be the acquiring banks and their subcontractors or even third parties (payment providers). A single third party may deploy a large number of CTPOS devices at different merchants or locations and manage them. Depending on the geographical location, these third parties may differ and it is not practically feasible for issuers to get in contact with all the third parties globally, in order to share secret keys with each other.

Furthermore, an authorisation entity sharing a unique cryptographic key with each individual CTPOS would be logistically impractical when considering the number of different acquiring banks, subcontractors, third parties and the sheer number of CTPOS devices out there in a global scale. After all, one of the objectives of introducing EMV, was to achieve the interoperability between different entities without prior business relationships.

We previously discussed the OPV mechanism used in ATM transactions. Taking the involved intermediaries into consideration and the need of providing confidentiality to the cardholder's PIN when it passes through one intermediary to the other, it can be assumed that a similar key-translation mechanism as in ATM transactions is used here. However, it is important to outline a notable difference which is that the ATM communication infrastructure (ATM Network) and the ATMs themselves are considered to be trusted.

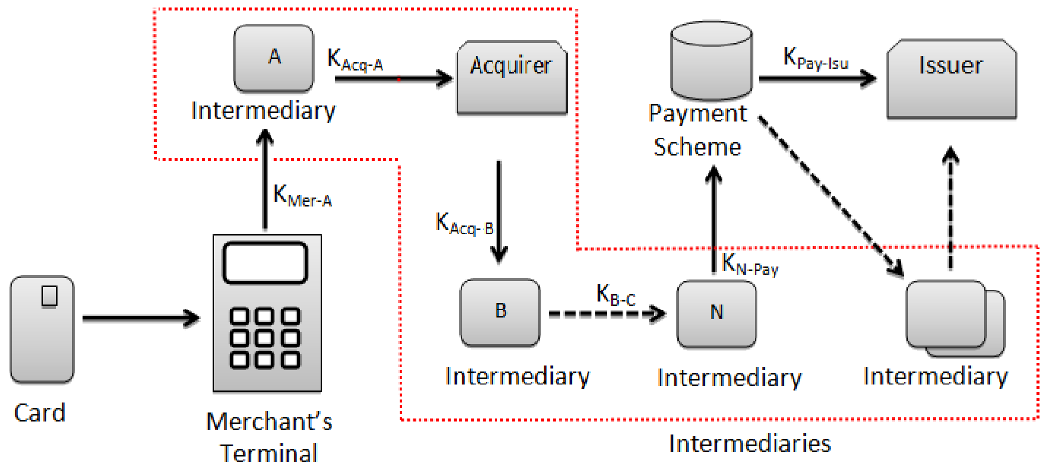


Figure 2.1: Online PIN Verification Message Flow

The Figure 2.1 illustrates the entities and the message flow in the OPV process. It shows how the involved intermediaries sharing unique cryptographic keys with each other engage in the key-translation process. Considering the communication link between intermediary A and the issuer for instance, the PIN block which contains the PIN is first enciphered by intermediary A using a shared key K_{Acq-A} before it is sent to the acquirer. Once received, acquirer deciphers the message to obtain the PIN block and subsequently enciphers it using the shared key K_{Acq-B} before forwarding to B. This process continues with each intermediary node until the message is received by the CIB or the authorisation entity.

2.1.3 Financial Fraud

According to a report published by the Financial Fraud Action UK (FFA) [93], the total cost associated with Card Not Present (CNP) fraud, such as: fraudulent purchases made using compromised card details, over the internet, telephone or by mail order amounted to £432 million in 2016. CNP fraud is also referred to as Remote Purchase fraud. The FFA report also outlines that the total international fraud losses on cards that had been issued in the UK was £200 million in the same year [93]. A proportion of the loss is due to Primary Account Number (PAN) compromise where counterfeit cards with compromised PAN details are used in countries that have not fully rolled out EMV Chip & PIN [150, 37].

PAN Compromise

Amongst the plethora of advantages provided by EMV, there is an inherent weakness associated with how PAN and related data is handled during a payment transaction in the current EMV architecture. Fraudsters and organised crime groups have exploited this weakness over the past years which has led to a significant rise in financial fraud [93, 150, 37, 174, 140, 181]. Regardless to whether it's a contact or contactless EMV transaction, in the current payment architecture, the PAN and related data is transferred in clear to the payment terminal. This is because the PAN and related data are processed in clear to complete a number of steps in an EMV PoS transaction [150, 14, 13, 15, 16, 24].

PAN data are compromised by fraudsters during EMV transactions or by breaching merchants' databases that store consumer payment card details [165, 30]. Fraudulently obtaining PAN and PAN related data to carry out financial fraud is also called PAN compromise.

Following a PAN compromise, the fraudsters may take different routes in order to make a profit, such as: selling the details on the black-market, making counterfeit cards or carrying out Cross-channel fraud. In Cross-channel fraud, the compromised PAN and related data, either during a PoS transaction or by breaching a database are used in other payment channels such as e-commerce payments.

To understand the scale of financial loss associated with PAN compromise the following case study can be taken as an example. The US retail giant Target was hacked by cyber criminals in 2013 [179]. The attackers used malware that infected PoS terminals at more than 1800 branches to compromised payment card details [165, 179]. The attackers compromised more then 40 million consumer credit/debit card details. The total financial loss for the company associated with the breach has mounted to around \$202 million. In May 2017, Target settled an \$18.5 million class action lawsuit related to the data breach.

2.1.4 EMV Tokenisation

In PAN compromise, the harvested PAN-related data is then used to carry out cross-channel fraud which includes Card Not Present (CNP) and Counterfeit Card transactions via internet, telephone or mail order. Tokenisation is increasingly being adopted by the payment industry as a method to prevent PAN compromise by mapping the PAN with a substitute value.

The process that manages the conversion from a PAN to a token and vice-versa is called tokenisation and the substitute value is called a token. Until recently, tokenisation was used by merchants and payment processing service providers to store consumer card details in a tokenised format to mitigate the risk of card data being hacked from databases [179, 30, 165]. This chapter does not refer to independent tokenisation methods used by various merchants and their payment processors in order to manage and store consumer payment card details in token format. The tokenisation discussed in this study refers to replacing the PAN used during EMV transactions with a token defined in EMV Tokenisation Specification [20]. The EMV Tokenisation Specification details requirements to support payment tokenisation in EMV transactions [20].

Following the standardisation of EMV tokenisation specification [20], there has been a dramatic move towards early adoption of this technology in contactless mobile payment applications. One example is the release of Apple Pay [22, 83, 25].

Tokenisation, discussed in this chapter, is a method that replaces the sensitive PAN used during an EMV transaction with a substitute value called the token. The token is a 13-19 digit numeric value that passes validation checks set by the payment scheme. A token is generated such that it does not reveal or conflict with the real PAN [20]. An adversary who captures the token cannot deduce the actual PAN.

EMV tokenisation is also in the interest of organisations that accept and handle consumer payments. In the payments industry, organisations, merchants, retailers, service providers, etc. that handle consumer payment card details are required to be compliant with industrial standards such as the Payment Card Industry-Data Security Standard (PCI-DSS). Compliance to such standards require adequate implementation of security controls, routine audits and management which can be costly and time consuming for organisations that accept and handle consumer payments. Therefore, storing and managing tokens instead of PANs in their servers and databases can simplify compliance audits such as PCI-DSS [155, 78, 39]. In addition to this, organisations that store customer payment details for faster payment checkout in subsequent transactions, can store tokenised payment details instead. This makes the payment details restricted to a specific merchant and unusable elsewhere, making it an unattractive target for cyber criminals.

2.1.5 Current Operating Environment of EMV Tokenisation

In this section, the current tokenisation online operating environment is presented. A generic payment architecture and the transaction message flow for a tokenised contactless mobile payment are shown in Figure 2.2 and explained according to [20, 83].

At the start of an online tokenised EMV contactless mobile payment transaction, the mobile device passes the payment token and token-related data to the terminal. The terminal forwards the authorisation request to the acquirer, who then forwards it to

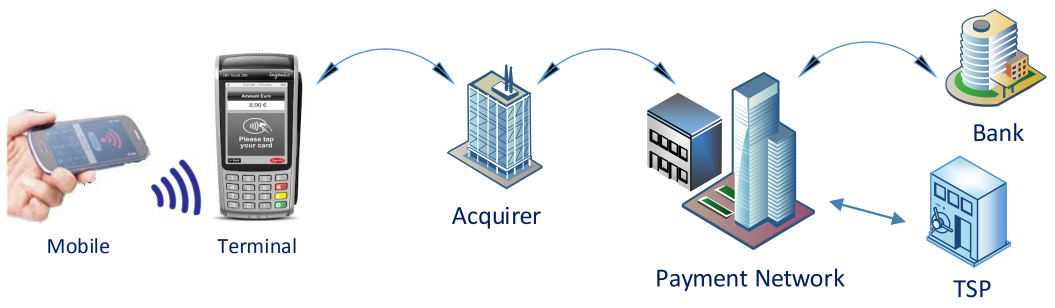


Figure 2.2: Generic EMV Tokenised Payment Architecture

the payment network. The entities involved in this process engage in a key-translation process, which means that two entities share a symmetric key to communicate, as shown by connecting arrows on both ends in Figure 2.2. When a message is forwarded to a particular entity, it is deciphered and enciphered using the shared symmetric key with the next entity. The payment network then communicates with the Token Service Provider (TSP) to de-tokenise the token in order to retrieve the PAN and validate the cryptogram [20, 83]. Following this, the payment network forwards the authorisation request with the mapped PAN details to the bank for authorisation. The bank, after carrying out necessary validations, sends an Authorisation Response Code (ARC) to the payment network in an authorisation response [20, 114]. The payment network now forwards the authorisation response to the terminal. The payment network may or may not send a TSP-generated response cryptogram in the authorisation response message to the terminal via the acquirer [20]. The terminal then approves or declines the transaction. As explained, the current architecture requires online connectivity during transactions.

In this section we discussed EMV based centralised payment technologies and their underlying payment architectures. In the next section, we introduce Bitcoin/blockchain based distributed payment technologies and discuss how these technologies differ from the centralised architecture.

2.2 Bitcoin/Blockchain based Distributed Payments

In the payment industry, digital ledgers are used to record details related to payment transactions. In the previous section, we discussed a number of payment technologies in the EMV based centralised payment architecture. In EMV based centralised payment technologies, digital ledgers are stored and shared in a centralised architecture. However, there are other payment technologies that uses different methods to store and share ledgers. In order to explain this clearly, we describe three architectures that can be used to share a digital ledger and identify their main differences. The architectures are illustrated in Figure 2.3.

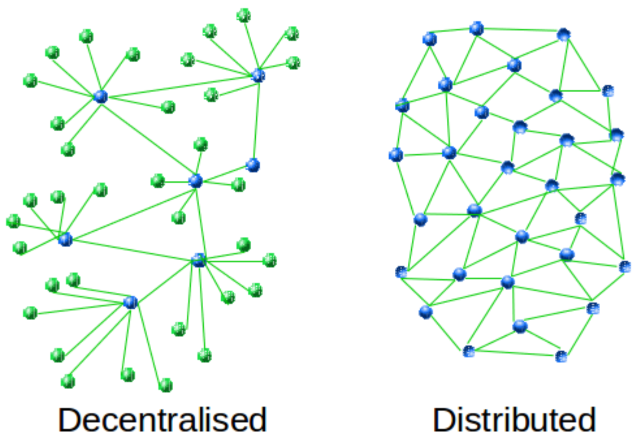


Figure 2.3: Architectures to share a Digital Ledger

A digital ledger that keeps a record of payment transaction details can be shared in three main architectures. In Figure 2.3, the nodes that store and share the ledger is highlighted in blue. The green nodes connect to a blue node to query the ledger. The significant differences of these architectures are listed below.

- Centralised: Client Server architecture with the server as one central node keeping the ledger.
- Decentralised: Distributed network of several centralised architectures connected by the central nodes. The central nodes share the same ledger and each central node serve a limited number of nodes. This way the network does not rely on a single server but uses a number of servers.
- Distributed: No central server keeping the ledger but each node is connected to various other nodes and all the nodes share the same copy of the ledger. Also considered a public shared ledger.

Distributed payment systems are payment schemes that are not controlled or managed by a single financial service, bank, payment network or a government. Instead, the responsibilities of managing the payment system is distributed to a large network of entities. Most of these payment systems work on a peer-to-peer basis and the nodes joining as peers help run the payment system. The concept of recording payment transaction related data in a shared distributed ledger is also called blockchain technology.

Unlike the fiat currency based centralised payment technologies that we discussed in Section 2.1, distributed payment systems are based on Cryptocurrencies which is a subset of what is generally known as Digital currency [71, 113].

A good example of this is Bitcoin which is a widely used distributed payment system to make anonymous payments [71, 147, 113]. Bitcoin is the first distributed cryptocurrency that was proposed in 2008. At the time of writing, there are hundreds of cryptocurrencies that have a market value [73, 71]. The common aspect of all these cryptocurrencies is the distributed ledger also known as the blockchain shared with participants in the payment system. Cryptocurrency falls under the category of distributed and/or decentralised Digital Currency which is not managed by a central payment scheme, financial institution, government, etc. but rather by the peers joining the distributed system [71, 113].

In the recent years, consumers, businesses, financial institutions, governments, etc. are starting to understand the benefits of using distributed payment methods to make online purchases and at PoS transactions [104, 142, 196, 67, 202, 56, 113]. Furthermore, a recent study estimates the current number of active cryptocurrency wallet holders to be between 2.9 - 5.8 million and the total cryptocurrency market capitalisation in April 2018 was $265 billion [104, 73].

In this section, we gave an introduction to distributed payment systems and discussed key features in regards to how these technologies operate. In the next section, we extend our discussion on blockchain technology and Bitcoin by investigating the underlying payment architectures.

2.2.1 Blockchain Technology

In the 1990's, peer-to-peer networks were gaining rapid interest and the technology was mainly used for the purpose of file sharing at the time. Peer-to-peer file sharing services such as Napster and Gnutella were used to share multimedia files such as MP3 [50]. Raising intellectual property and copyright concerns, unregulated peer-to-peer file sharing services were classed as earlier forms of darknets [50].

Just after the year 2000, innovations based on peer-to-peer networks found new ground other than file sharing [177, 50]. One such example is the invention of the Bitcoin peer-to-peer network [147, 52].

The blockchain technology as we know of today, is one of the significant innovative outcomes that came from Bitcoin [147, 71, 113]. Bitcoin is a distributed cryptocurrency system that works on a peer-to-peer network. Unlike in a banking networks where transactions are approved by dedicated servers, in distributed payment systems, all transactions are approved by peers joining the peer-to-peer network. Transactions are represented in hash outputs and the ownerships of monetary value attached to a public key (also known as the coin address) is transferred using digital signatures. Transferring ownership of the value attached to a coin from one user to the other using digital signatures creates a chain of signatures. The recipient can verify the ownership of a coin using the signature chain [147]. The recipient, however, cannot check whether the sender has already transferred the coins to another user and is trying to make a duplicate payment with the same coins. This is referred to as the double-spending problem and a payment system should support detection of any attempted double spending. As there are no dedicated server or an authorisation entity to verify each payment transaction, distributed payment systems solve the double spending problem is by introducing a shared distributed ledger that permanently records all transactions that have completed in the past [147, 113]. This innovative solution is also called the blockchain. To provide a permanent record and an audit trail, past transactions are included in blocks. A block is a record of sequences of digitally signed and verified transactions. This provides a permanent record of a sequence of blocks from the very first to the latest block which can be used by peers to prevent double spending. The created blocks are timestamped and chained together in the order they appeared, deriving the term blockchain [147, 113]. A blockchain is shared and synchronised with all the nodes connected to the peer-to-peer network in a global scale. The blockchain can be referred to as a globally distributed cryptographic ledger.

Types of Blockchains

Blockchains can be categorised in to two main categories based on permissions, who can recorded information on the blockchain and who can view recorded information, etc. The two main types of blockchains are described below.

1. **Unpermissioned:** A unpermissioned blockchain is a distributed ledger shared and synchronised with all the entities in a peer-to-peer network. The management of the ledger, how and when things are recorded on to the ledger is not controlled by a single entity. The ledger is visible to all the entities and any entity on the peer-to-peer network can append or record a transaction as long as it is genuine. The integrity of the ledger is agreeing upon by using a majority consensus. The Bitcoin blockchain is a good example.

2. **Permissioned**: A permissioned blockchain is a distributed ledger that has one or a given number of owners. Any records to the ledger can be only added by these owners and not by any other nodes. The consensus in regard to the integrity of the ledger is reached only by entities with pre-defined privileges. Due to the limited

consensus, transactions are much faster compared to unpermissioned blockchains, the owners have more control over the consensus mechanism and management of the blockchain.

In the next section, we extend our discussion on Bitcoin, explain its underlying blockchain based architecture.

2.2.2 Bitcoin

Bitcoin is a distributed digital cash system based on a peer to peer network architecture. Satoshi Nakamoto which is a pseudonym (online identity - might not be the real name) proposed and developed the payment system in October 2008. In January 2009, the first hash block called the Genesis Block was created and a publically available global ledger called the Blockchain was broadcast on the Bitcoin peer to peer network. There after, the Bitcoin payment system is widely used to make anonymous payments over the Internet. Payment transactions are broadcast to all the nodes in the peer to peer network and are permanently recorded. Bitcoin transactions are represented in SHA 256 hash outputs. There are two types of broadcasts; Transactions and Blocks. The system is built to have a fixed number of Bitcoins in circulation; this amount is roughly 21 Million Bitcoins. As of April 2018, there are about 16.9 Million Bitcoins in circulation [55]. In Bitcoin mining is the process a group of nodes authorising transactions and creating new blocks. On average, in the Bitcoin network a new block is created every ten minutes. Verification of transactions is assigned randomly to network peers and the verifying peers get a small portion of Bitcoins. However, the creation of a valid new block pays the creator 12.5 Bitcoins at the current level which is also known as Coinbase within the Bitcoin community [53, 52]. This amount halves every four years to balance out the increase of computational power over time. A user can start using Bitcoins by running Bitcoin wallet/client software. It can be used to manage Bitcoin addresses/keys and to make payments. Every user needs to have a Bitcoin address to receive Bitcoins from other parties and a corresponding Secret Key is needed to transfer Bitcoins to other parties.

A user can generate a unique Bitcoin address which is also called a Bitcoin public key by using an Elliptic Curve Digital Signature Algorithm (ECDSA) key as given below [53, 147].

Step 1: Generate Private ECDSA key. ***Step 2:*** Generate Public ECDSA key.
Step 3: Create a SHA-256 hash of step 2. ***Step 4:*** RIPEMD-160 hash of step 3.
Step 5: Add bytes 00 to the front of step 4. ***Step 6:*** SHA -256 hash of step 5.

Step 7: SHA -256 hash again of step 6. ***Step 8:*** Take the first 4 bytes of step 7. ***Step 9:*** Add these 4 bytes at the end of step 5. ***Step 10:*** Base58 encoding of step 8.

Each Bitcoin public key has a corresponding private key which is used by the payer to generate a digital signature when making a payment.

Bitcoin works in a slightly different way from other digital currencies by having a globally broadcasted transaction record that links ownership to a Bitcoin instead of a unique string that represents a coin [69, 163, 151].

A Bitcoin transaction is the process of transferring the monetary value attached to a particular Bitcoin address by digitally signing a hash of a previous transaction together with the next owners public Bitcoin address and adding this record to the shared ledger. The ownership is transferred from one address to the other by using this chain of signatures. This links past transactions to the present ones.

The transaction verification and creating new blocks by the Bitcoin miners is made a fair and non-trivial task by introducing a *Proof of Work* method, a concept first introduced by Adam Back in 2002 as a counter-measure against unsolicited junk mail and denial of service attacks [42]. Peers who engage in this process are called Bitcoin miners and miners are rewarded for their computation. This concept was implemented by Bitcoin making the peers to generate a hash less than a target value with a certain number of zero bits at the beginning of the hash. Generating a hash that has a certain number of zero bits at the beginning is a difficult task that takes computational time. The work needed increases exponentially as the required number of zero bits in the beginning increases. As a result, a constructed block cannot be easily changed [53, 147].

Based on the consensus mechanism, nodes in the network accept the longest blockchain. Because of this, to include a manipulated block the attacker needs to re-perform the proof-of-work for the modified block and all subsequent blocks that appeared chronologically. This is considered practically infeasible.

The need for having continues relationship with past transaction hashes to the present ones incurs the need for storage space to store all the hashes. The system over comes this drawback by making the transactions hashed in a Merkle Tree [139]. This makes the hash only needing to record the Root Hash [147, 139].

Therefore, Bitcoin transaction security relies on the correctness of the Block Chain [147, 53]. By looking at one's Bitcoin address, the true identity of the user cannot be revealed. However, due to the necessity of having to broadcast all transactions publicly prevents the anonymity of Bitcoin payment transactions. This has become a drawback and with the advancements in computational power and data analysis, it may be possible to link Bitcoin transactions to real user identities [164, 44, 170, 203, 124]. There have been several attempts to improve the security of Bitcoin recently. These include techniques to address attacks on Bitcoin [44] and solutions to address anonymity, such as Zerocoin which is a proposed distributed digital cash system that acts as an extension to Bitcoin [141]. Furthermore, MasterCoin is a proposal to make Bitcoin more stable and secure by adding a new protocol layer on top of Bitcoin to run services [198].

2.2.3 Existing SMS Payment Systems and Bitcoin

In developing countries, SMS (Short Message Service) mobile payment systems have been extremely successful. One such example is the popular M-PESA in Kenya [173].

The SMS approach has been extended to perform Bitcoin transactions e.g. Bitcoin Currency (BTC) For SMS [28], where users can add Bitcoin to their online wallets using SMS, and Coinapult Bitcoin SMS service [2] which a range of Bitcoin transactions can be done[1]. However, the problem with these schemes is that all requires the users to have online access to set up the Bitcoin wallets and to maintain them.

There have been attempts to integrate Bitcoin with SMS based payment systems such as M-PESA in Kenya but failed because of business pressures [23, 205, 1]. However, Bitwala offers a Bitcoin remittance transfer service to Uganda, Tanzania and Nigeria that uses mobile money services [27], but again, the user needs online access to their Bitcoin Wallet.

Other proposals to carry out Bitcoin transactions using mobile phones require the users to download smartphone apps to interact with online Bitcoin wallets e.g. BTC Wallet [96], which is not feasible for the environment under discussion, both in terms of equipment available and online access.

2.3 Anonymous and Fair-exchange Payment Schemes

In the previous section, we introduced Bitcoin. Anonymous payment schemes such as Bitcoin helps realise anonymity and user privacy during payment transactions are called *Anonymous Payment Protocols.* Digital Cash is a variant of anonymous payment protocols [113]. Protocols that are built to achieve fairness in e-commerce transactions are called *Fair-exchange Protocols.* A combined solution that would realise fairness as well as anonymity is called an *Anonymous Fair-exchange Payment Protocol.* In the next two sections, we carry out a discussion to identify these schemes.

[1]The Coinbase SMS service was discontinued in March 2017 in favour of their smartphone apps [148].

2.3.1 Anonymous Payment Schemes

Anonymous Payment Protocols provides anonymity to the payer during a payment and improves privacy of the payment transaction. There are two main types of anonymous payment systems: on-line payment systems and off-line payment systems. In an *on-line scheme* the payer, payee and the bank requires an online connection at least once during the protocol to verify the digital cash. The first on-line anonymous payment system was proposed in 1985 [68]. In 1989 a new payment scheme was introduced that uses a special type of user account called ‘standard values’ to achieve anonymity [62]. Supporting this, a new scheme that considerably reduces the size of these databases was proposed in 1994 [63].

The significant aspect in *off-line schemes* is the fact that the Trusted Third Party (TTP) does not have to be on-line during the protocol run between the payer and the payee. Instead, the TTP verifies whether digital cash has been double-spent when the payee presents the digital cash to the TTP to be deposited in to his/her account. Chaum et al. proposed one of the first off-line anonymous digital cash schemes in late 1980’s [69]. Ferguson scheme is a digital cash system that falls under the category of transferable cash [92]. DigiCash was a real world implementation, founded 1990 by Chaum. However, the company did not succeed. CyberCash founded in 1994, provided a wallet application and a micropayment system called CyberCoin designed with the use of Netbill protocol [154]. Mondex is a digital cash system that is based on Smart Card technology. The project was initiated in 1991 and at present Mondex is part of the MasterCard worldwide suite of smartcard products. Mondex at first achieved a good start, but could not expand as expected in to wider deployment [132].

2.3.2 Fair-exchange Schemes

E-commerce transactions most of the time involve buying and selling of electronic content, goods and services between parties that have no prior trust-relationships with each other. This requires e-commerce payment, contract signing, digital content exchange, certified delivery protocol schemes and implementations to improve fairness between involved parties. Fairness in e-commerce can be categorised as Weak and Strong fairness.

Weak fairness is when in an electronic transaction between two parties, the honest party can prove to a third party after the transaction protocol-run, that he/she followed the protocol even though the dishonest party did not send or pay to the honest party or aborted the protocol [162].

On the other hand, Strong fairness or True fairness ensures that, the protocol itself tries to avoid disputes and misbehaving of parties and resolve any disputes within the protocol without reaching an external judge. These protocols make sure that due to the misbehaviour of a dishonest party, a honest party engaged in a transaction does not get penalised [40]. Due to this, either both parties are guaranteed to receive their items or none of them receive anything.

When it comes to e-commerce scenarios such as exchanging electronic content, only a few Anonymous Fair-exchange Protocols have been proposed [109, 208, 209, 207]. Depending the involvement of a trusted third party, fair-exchange protocols can be divided in to two main categories.

Two party based protocols (Gradual-exchange)

These protocols do not rely on a TTP to achieve fairness but employ a process of gradual exchange of several messages between the transacting parties to augment the probability of fairness over the rounds of exchanged messages. Proposals include a protocol based on bit-by-bit information exchange [58] and a system called the *"1-out-of-2 oblivious transfer protocol"* [162]. In the above protocols, the party with the most computational power than the other could gain advantage by conducting brute-force attacks on received message to compute the remainder [47]. In 1990 a probabilistic protocol was proposed as a solution [47]. However, the protocols discussed above lack simultaneity of exchange and during the protocol, involved parties could misbehave for their own advantage [162].

Protocols based on a Trusted Third Party

TTP based protocols can be classified into three main types depending on the TTP's involvement.

In-line TTP based protocols, involve the TTP to collect exchanged items, check their accuracy and finally forward them to the intended parties. Proposals include, the Believers Protocol which guarantees confirmation of sending and receiving of certified electronic mail [43] and a protocol which adopts a TTP as a non-repudiation server [72]. Strong fairness is provided because of the involvement of the TTP. However, to manage and maintain large amounts of communicated messages, the TTP is required to be readily available any-time which is considered a bottleneck.

On-line TTP based protocols have proven to be more efficient than in-line TTP protocols due to the fact that, the TTP involves in the protocol run but not in every transmitted message. However, the involved TTP still engages in the protocol run to guarantee fairness by validating, storing and generating transmitted messages. The

Netbill Protocol was one of the first protocols which achieved strong fairness [79]. Another protocol was proposed aiming at Electronic Data Interchange (EDI) systems to achieve non-repudiation [206]. Zhou and Gollmann proposed another on-line protocol which tried to reduce the workload of the TTP [210]. The protocol was designed to provide evidence for the sender and the receiver, during the protocol run as well as after completion. Zhang et al. proposed another protocol which would provide fair-exchange, user anonymity and privacy payments over the internet [209].

Off-line TTP based protocols lets the transacting parties exchange products without the involvement of a TTP unless transacting party misbehaves, prematurely aborts or a communication failure happens. These protocols are also called *"Optimistic Protocols"*. The first optimistic protocol was proposed in 1989 [62]. An improved protocol which allows exchange of two digital content between two parties was another proposal [41]. The TTP in this protocol gathers enough evidence of a dishonest party to present in front of a judge. Furthermore, an Ambiguous Optimistic Fair Exchange Protocol without Random Oracles has been proposed recently [106].

In this section, we carried out a background research on anonymous and fair-exchange payment protocols. In the next section, we introduce formal analysis tools that can be used to verify the security of cryptographic protocols.

2.4 Formal Analysis Tools

This section introduces formal analysis tools. Evaluating the security of a proposed protocol is important to identify potential weaknesses and to improve the protocol by addressing the identified weaknesses. The security requirements of a cryptographic protocol can be evaluated using a formal analysis tools. These tools check for network related attacks in the communication channel between the sender and the receiver. Even though formal analysis is not the main focus of this thesis, we use it to show that the protocols proposed in this thesis are secure and complete.

There are a number of formal analysis tools, such as: AVISPA, ProVerif, CasperFDR, Tamarin and Scyther. Comparisons between a number of these tools are carried out in [85, 82]. The comparisons in [85, 82], identified that, performance wise, ProVerif was the fastest tools and Scyther came in close second but had the advantage of not using approximations. Furthermore, Scyther also supported handling unbounded verifications. The comparisons also found that, CasperFDR had an exponential behaviour and was faster then AVISPA's implementation called SAT-based Model Checker tool (Sat-MC) for a smaller number of protocol runs. Furthermore, AVISPA was found to be slower then other analysis tools and had the drawback of not being able to display an attack tree [85, 82]. We use CasperFDR and Scyther for formally analysing our protocols in this thesis. Our choice of these two tools were mainly based on the usability aspect. Both tools provide a Graphical User Interface (GUI), ease of modelling the protocol and clear outputs. From the two selected tools, Scyther provides verification of synchronisation as an additional security claim of a given protocol. Furthermore, we found Scyther to be user friendly and provided easy installations for Microsoft, Linux and Mac operating systems.

The combination of CasperFDR and Scyther used in this thesis for mechanical formal analysis are sufficient to verify the security claims of the proposed protocols. More specifically, the tools are capable of analysing security claims such as: Confidentiality, Integrity, Aliveness *(Alive)*, Weak agreement *(Weakagree)*, Non-injective agreement *(Niagree)* Non-injective synchronisation *(Nisynch)*, Secrecy of data *(Secret)*, etc. under the Dolev-Yao adversarial model in associated protocols [87, 81, 80].

Another type of alternative formal analysis technique is called Probabilistic Model Checking, mainly used for analysing quantitative properties such as: reliability, responsiveness or resource usage [127]. Tools developed to carry out quantitative analysis using probabilistic model checking include: PRISM [126, 128] and The Markov Reward Model Checker (MRMC) [120, 119]. These tools are used for analysing quantitative properties of systems that exhibit probabilistic behaviour such as: communication loss in a wireless channel, identifying the probability of a component of a system (e.g airbag, brakes, network sensor) failing, etc.

Our main objective of the thesis is to enhance the security of payment transactions. Quantitative analysis is not the main focus of the thesis. Therefore, the selection of formal analysis tools were based on the capability of verifying the security claims of the proposed protocols. The CasperFDR and Scyther mechanical formal analysis tools used in the thesis are suitable and capable of verifying the security requirements.

2.4.1 CasperFDR

The security requirements of a cryptographic protocol can be analysed using the process algebra Communicating Sequential Processes (CSP) introduced in [105] to produce a description of a protocol system and its model checker Failures-Divergence Refinement (FDR) [171]. As the process of producing a description of a system in CSP is time consuming, a combined solution which is called CasperFDR was proposed in [129]. CasperFDR uses the Dolev-Yao threat model in [87] for the security verification of protocols. In Dolev-Yao threat model, the adversary is assumed to have full control over the communication channel where, the adversary has capability to impersonate other users, intercept and modify communicated messages.

The security protocol can be modelled as an abstract using a text editor and provided as input to Casper. The tool then produces a CSP description of the protocol in a .csp file. Then the .csp file can be verified by FDR which outputs potential attacks and weaknesses if detected. CasperFDR uses *Running* and *Commit* events to verify the authenticity of communication between two entities. The *Running* is executed by the receiver after receiving a message by the sender and the *Commit* is executed by the sender after receiving a reply by the receiver. When verifying security requirements, CasperFDR uses the *Claim* event. For example, the secrecy of communicated data can be verified using *Claim_Secret*. This makes a protocol-run between the sender and the receiver to complete.

When verifying the security of a protocol using CasperFDR, an input script that has two main parts of the modelled protocol needs to be submitted. The first part of this script provides the protocol definition. In more detail, this defines the communicating entities and their initial knowledge of components related to the protocol messages. *#Free variables* is used to define users, variables and functions used in the protocol. *#Protocol description* is used to model the protocol messages, their content and the sequence they are transmitted. After this, the *#Processes* declares the roles of different entities used in the CSP process (INITIATOR, RESPONDER, SERVER) and provides their initial knowledge (using the term 'knows') of values at the beginning of the protocol. The next input in the script is *#Specifications* which defines the security requirement such as: *agreement* for authentication and *secret* for confidentiality of the analysed protocol.

The second part provides the system definition. In more specifically, this defines the actual entities in the system, attackers knowledge and capabilities. The *#Actual variables* also referred to as *#Type definitions* defines the names of the entities to be used in the FDR verification. They are similar to the ones listed in *#Free variables*. Additionally the Intruder is also defined as a participant in the protocol. Next, the functions such as public/private keys used by the entities of the protocol are defined using *#Functions*. The *#System* is used to declare the entities of the protocol with their actual names and knowledge. Finally, the name of the intruder and the intruder's knowledge of other entities in the protocol and values communicated are declared in *#Intruder*.

2.4.2 Scyther

Scyther is a formal analysis tool that can be used to verify the security requirements of a cryptographic protocol [80, 81]. Scyther is capable of analysing a protocol for an unbounded number of instances compared to other formal analysis tools in [85, 82]. The

Scyther tool can be installed in a number of operating systems including: Microsoft Windows, Linux and Mac OSX, where as CasperFDR only supports Linux. The Scyther tool has a user friendly GUI with menu tabs to run verifications from an input file, read help information and change settings related to set the verification parameters. Scyther mainly uses the Dolev-Yao threat model in [87] for the security verification of protocols. However, Scyther can support other threat models.

The description of a protocol can be modelled and provided as input to Scyther using the Security Protocol Description Language (SPDL) defined in [81]. In the input .spdl file, the functions, variables and constants of the protocol are declared at the beginning. The *usertype* command generate new data types. The spdl provides three main protocol modelling features: *roles, events and claims.* The entities in a protocol are described using a set of roles, which characterise events. The send and receive operations are classed as *send* and *recv* events respectively; each corresponding *send* and *recv* event has the same sequence number.

In the script *roles* is used to declare entities communicating in the protocol and a *role* can execute multiple protocol-runs. The variables used to store received values are defined in *Var* and newly generated parameters are declared by *Fresh*. The *Macro* can be used to replace a set of values that are repeatedly used in a protocol by an abbreviation. The *Ticket* can be used to replace an unknown value.

The security requirements and objectives of a protocol that require verification are specified using *claim* events. More specifically, the secrecy of data is verified using *secret*. Security requirements related to authentication are verified by: *Alive* for aliveness of a protocol run, *Weakagree* for weak agreement, *Niagree* for non-injective agreement and *Nisynch* for non-injective synchronisation [81, 80]. Aliveness means that, a receiver successfully completes a protocol with a sender, then the sender has previously been running the protocol. Non-injective synchronisation means that the send and receive events happens to the expected sequence. Non-injective agreement means that the communicating entities agree with the communicated content at the end of the protocol run.

After verification, an output summary is provided. If a potential attack is identified by Scyther, it outputs a graph detailing the attack in addition to the summary.

2.5 Summary

In this Chapter we first presented the background research related to EMV based centralised payment systems. More significantly, we introduced aspects in the EMV payment architecture such as: EMV OPV process, Tokenisation and operating environments.

Following this, the background related to Bitcoin/blockchain based distributed payments was presented. Under this category, we gave an detailed introduction to Bitcoin and blockchain technology. In both the centralised and distributed payments discussed in this chapter, we investigated the main features/aspects inherent in each payment architecture and explored their operating environments.

After this, we discussed anonymous and fair-exchange payment protocols. We carried out a discussion to explain why these protocol are important in e-commerce based payment transactions. Finally, we introduced mechanical formal analysis tools and how these methods can be used to evaluate the security of our proposed protocols.

Part II

Improvements for EMV based Centralised Payments

Chapter 3

Improving Online PIN Verification Security

Contents

In this chapter, we present our first contribution of the thesis under the EMV based centralised payment category. We first discuss the EMV Online PIN Verification (OPV) process and explain how OPV can be deployed in two methods (Segmented and Unified Authorisation) in the current EMV payment architecture. We then reflect upon the indelible trust assumptions placed on the intermediaries in the EMV payment architecture. In this chapter, we focus our attention on the Segmented Authorisation method and identify potential attack scenarios that can compromise the security of the OPV process when these trust assumptions are scrutinised. Addressing these potential issues, we then propose a number of solutions that improve the security of OPV process in Segmented Authorisation method.

3.1 Introduction

In Section 2.1.1, a brief introduction to EMV and its architecture was given. The PIN verification process in an online authorisation environment is called the Online-PIN Verification (OPV). An introduction to OPV is given in Section 2.1.2. Investigating the OPV process is the main focus of this chapter. In our investigation, we elaborate on the indelible trust assumptions placed on the intermediaries (subcontractors) between a payment terminal and the scheme operator/card issuing bank. This trust assumption, makes rest of the participants in the payment architecture assume that the intermediaries are trusted and secure. When this trust (assumption) is scrutinised, we discuss a potential attack scenario that can be used by a financial fraudster to compromise PIN related information.

We further discuss how this information can be used by the fraudster to carry out an online PIN approved transaction without the involvement of the genuine cardholder but with the correct PIN. We then propose three solutions based on the existing OPV process that defend against these attacks. The proposed improvements are then implemented to measure any incurred performance penalties. In our practical implementation as detailed in section 3.4.2, however, we implement and measure the performance penalties of both symmetric and asymmetric encryption methods proposed in each of our solutions. The results of performance measurements are included in Tables 3.7. Finally we subject the proposed protocol to mechanical formal analysis using CasperFDR to evaluate the security guarantees.

In Section 3.1.2 we explain why securing the PIN is vital for all the stakeholders at different levels of the payment architecture. To secure PIN and associated transaction details, modern payment terminals are designed to be secure and tamper resistant [159, 13, 133].

In section 2.1.1, we explained how EMV supports offline and online PIN verification methods. In the offline PIN verification, the PIN is verified by the payment card and in OPV the PIN is verified by the authorisation entity. The PIN verification carried out by the authorisation entity is considered as having a higher assurance level then the PIN verified by the payment card. This is because the CIB as the entity that issues the payment card also sets the PIN and manages any PIN changes for each particular customer. The main focus of this chapter is the PIN verification carried out by the authorisation entity which is referred to as OPV.

The financial institutions especially the banks suffer significant financial loss due to payment card fraud [93, 150, 37, 174, 140, 181]. The main focus of this Chapter is to improve the security of OPV process which would help to prevent potential attack scenarios that we identify in our work. Preventing such attack scenarios would significantly help improve the security of payment transactions and reduce the risk of potential financial cost associated with such attacks if they are realised.

In the next section, we extend our discussion on EMV OPV and explain two different methods how OPV process can be deployed in the current EMV payment architecture.

3.1.1 Two OPV Deployment Methods

In this section, we discuss the two different OPV methods that may be used in the current EMV payment architecture. In our discussion, the two OPV methods are differentiated based on whether OPV is carried out together with the online transaction authorisation or as a separate authorisation. Then we explain how the OPV process can be carried out at different stages of the overall EMV transaction, depending on the selected deployment method.

Segmented Authorisation Method

In this section, the OPV process discussed in Chapter 3 is revisited. As explained previously, in the Segmented Authorisation method, the OPV is carried out separately to the online transaction authorisation. In order to explain this more clearly, we list the transaction steps in this deployment method as follows:

1. The CTPOS first constructs the encrypted PIN block which contains the user's entered PIN.

2. The enciphered PIN block is then sent to the authorising entity to be verified. The authorising entity could be the Scheme Operator (SO) or the Card Issuing Bank (CIB).

3. If the PIN is verified correctly, the authorising entity sends a response message back to the CTPOS confirming that the OPV was successful.

4. It is after this confirmation of the outcome of the OPV, that the CTPOS proceeds to the online transaction authorisation.

5. The online transaction authorisation request contains the card generated Authorisation Request Cryptogram (ARQC).

6. The card generates a cryptogram and forwards it to the CTPOS.

7. The CTPOS sends the cryptogram in an online transaction authorisation request to the authorising entity.

8. The authorisation entity verifies the cryptogram and sends a response message to the terminal indicating whether the transaction was approved or declined.

As we can see, in the Segmented Authorisation method, the OPV process initiated and completed before the transaction authorisation. This separation of the two processes raised a few security concerns that we discuss in subsequent sections. The proposed solutions that addresses the potential attack scenarios related to the Segmented Authorisation Method was previously presented in Section 3.3.

Unified Authorisation Method

In this section, we introduce the second deployment method which is different to the one we discussed above. The differentiating factor is mainly related to the communication sequence of the OPV and the transaction authorisation. In contrast to the Segmented Authorisation method, in the Unified Authorisation method that we introduce here, the OPV is carried out together with the online transaction authorisation. To explain this more clearly, we list the transaction steps in this deployment method as follows:

1. Similar to the previous method, the CTPOS first constructs the encrypted PIN block which contains the user's entered PIN.
2. Instead of sending the enciphered PIN block to the authorising entity separately, in this instance the CTPOS requests the ARQC from the card.
3. Once the ARQC is received, the CTPOS constructs a unified message which includes the OPV and the online transaction authorisation requests.
4. The online transaction authorisation request contains the card generated ARQC.
5. The ARQC together with the enciphered PIN block is then sent to the SO/CIB.
6. After receiving the request messages, the SO/CIB conducts OPV and online transaction authorisation. The outcomes of both verifications are then sent back to the CTPOS in a response message.

As we can see, in the Unified Authorisation method, the OPV process and the transaction authorisation is carried out together in the same message sequence. In this section, we explained both the Segmented and Unified Authorisation methods. In the next section, we explain a potential issue that raise concerns regarding the security of OPV process.

3.1.2 Problem Statement

In the current EMV architecture in which OPV process is carried out, there are a number of intermediary entities in the communication path between the CTPOS and the SO/CIB. The EVM Card Specification Book 4 recommends and outlines why this communication should be adequately protected [16].

By examining the OPV process used in the ATM transaction architecture [195] and referring the literature in [49, 131, 65], it is clear that first the CTPOS requests the PIN from the cardholder. Once the PIN is received the CTPOS encrypts it before forwarding the encrypted PIN details to the authorisation entity for verification.

During a transaction, the CTPOS encrypts the PIN using a symmetric key shared between the terminal and the first point of contact (which is not necessarily the SO/CIB) in the communication channel towards the SO/CIB. It must be noted that the symmetric key used to encrypt the PIN between the CTPOS and the first point of contact can also be a session key. We explain this process which is also termed as the key-translation in Section 2.1.2. The same key-translation mechanism is used by the first point of contact to forward the PIN to the next entity in the communication path towards the SO/CIB.

We explained the EMV OPV process in section 2.1.2. There is another process in EMV called the “transaction authorisation” which is used to decide whether a particular transaction can be authorised or declined. As part of this process, during a transaction the payment card generates a message to request authorisation to a particular transaction by the authorization entity which is called the transaction authorisation message. In the EMV specification, this authorisation request is called the ARQC. However, the current payment process has no binding between the ARQC and the OPV process [14, 13, 15, 16].

The ARQC is encrypted by a symmetric key shared between the payment card and the authorisation entity. The encrypted fields include a number EMV tags that are standardised parameters in the EMV specification. In EVM Card Specification Book 4, it details that one of these tags is the Cardholder Verification Method (CVM) which indicates the method used to verify a cardholder during the transaction [16]. Normally the cardholder verification is carried out before the transaction authorisation.

The CVM is a tag that is three bytes in length. The bytes indicates: the CVM performed, CVM conditions and CVM results [16, see: p49]. When it comes to information related to the OPV process in the CVM tag, the only information related to the OPV is a single binary value that is set to 1 if CVM was performed at the start of a payment transaction [15, see: p162]. As we can see in both the CVM or the ARQC, there are no tags/parameters that binds the OPV process with the associated ARQC.

Another key observation that we understand is that during an OPV, the PIN is handled (creating a PIN block to be sent to the authorising entity for approval) only on the PIN entry device which in this case the CTPOS and the payment card is not informed about the PIN value entered on the CTPOS.

The risk associated with the weaknesses and shortcomings that we briefly outlined above is realised if an adversary compromises one of the intermediaries that engage in the key-translation process between the CTPOS and the authorisation entity. Such compromise would lead to the adversary obtaining details such as PIN and associated payment transaction related information to carry out further financial fraud.

The adversary at the compromised entity will be able to observe OPV messages that include PINs and associated Primary Account Numbers (PAN). By obtaining such information, the attacker is now in a position to carry out an OPV-based transaction at a merchant's premises with a stolen card for which the adversary has previously obtained the relevant PIN. This way an attacker can perform an EMV transaction that requires online approval at a CTPOS.

As indicated in reports [45, 117, 118], the notion of an attacker being able to compromise an intermediary in the payment architecture is not hypothetical or too far-fetched. It is important not to underestimate such scenarios for the overall improvement of the payment system.

3.1.3 Contributions

In this chapter we propose solutions that enhance the security of EMV OPV process by addressing the aforementioned security concerns that we briefly identified. Our main contributions for this chapter are two fold. The first contribution is the proposed enhanced OPV process that proposes three separate solutions to protect the PIN details. The second contribution is how we bind the OPV with the respective ARQC in an online transaction authorisation scenario.

1. To protect the PIN we have proposed an enhanced OPV process using:
    (a) Card-based solution with symmetric cryptography.
    (b) Card-based solution with asymmetric cryptography.
    (c) Payment terminal-based solution with asymmetric cryptography.
2. Binding each OPV with the associated ARQC.

The remainder of the chapter is structured as follows. In section 3.2 the payment networks operating environment is outlined and the attacker's capabilities and potential attack vectors for compromising the OPV process is discussed. In section 3.3, the three potential countermeasures that address the security concerns are provided. The proposed solutions are then analysed in section 3.4. Finally, concluding remarks are provided in section 3.5.

3.2 Potential Concerns

In this section, we discuss the operating environment and make assumptions related to the OPV process. Following this, we introduce our attack model, explain the attacker's capabilities and examine two potential risk scenarios.

3.2.1 Operating Environment and Assumptions

The operating environment of the OPV process is illustrated in Figure 3.1. As we previously discussed, the PTO can represent a number of entities such as the acquiring banks, subcontractors or third parties. In Figure 3.1, these intermediaries are outlined with a dotted rectangle. In OPV, the communication path between the payment terminal and the authorisation entity (i.e. SO/CIB) may consist of numerous intermediaries such as: the acquirers' subcontractors who operate CTPOS devices, third party PTO and other entities that engage in key translation (where messages are enciphered and deciphered from node to node).

A PTO might issue or lease their payment terminals to a number of customers (merchants). In our operating environment, however, we assume all the scenarios: the PTO is a third party that manages the terminals, subcontractor or it is an acquiring bank. The choice of who is the PTO does not affect our operating environment and the risk scenarios discussed later. It must be noted that it is more likely that there can be some additional nodes between the PTO and the scheme operators. These additional nodes can be a number of entities in the communication path to the SO. In the event of the PTO is a third party, one of these additional nodes must be the acquiring bank.

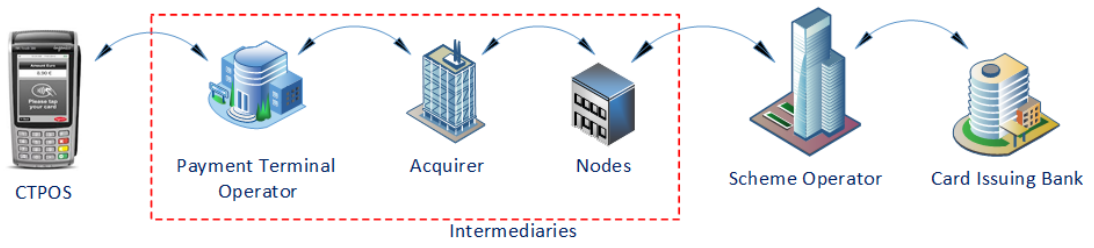


Figure 3.1: OPV Operating Enviroment

As illustrated in Figure 3.1, the SO is directly connected with the CIB and has a secure communication channel. The arrows connect two entities with each other and the arrowhead indicated the direction of the message flow. Furthermore, each arrow connecting two entities uses a unique session key to secure the communication between those two entities.

The key-translation process is performed by each intermediary and SO until the OPV message is received by the CIB. In some scenarios, the SO performs the OPV verification on behalf of the CIB. However, in this chapter we are not considering such scenarios. However, the risk scenarios that we discuss in this chapter is not affected by this. The terminal, each intermediary node and scheme operator will perform key translation until the OPV message is delivered to the CIB.

The following entities are considered to be trusted and secure in this study: the smart cards, payment terminals, SO and the CIB. The rationale for this consideration was discussed in Section 3.4.1. However, any intermediary that might be connected to the public internet is considered to have the potential to be compromised. This assumption can be considered reasonable by referring to past incident reports [45, 117, 118, 54, 110, 125]. In one incident attackers were able to successfully infiltrate sensitive information from the banking sector [118].

Here, we gave an overview of the operating environment. In the next section, we outline the capabilities of the attacker after taking into consideration the operating environment, the assumptions made about the intermediaries and other entities in the payment architecture.

3.2.2 Attacker's Capability

The capabilities of the adversary are listed below:

1. Has the capability to compromise any of the intermediary nodes.
2. Has the capability to access the OPV communication in plaintext on the compromised intermediary node. As in the operating environment discussed in section 3.2.1, individual intermediary nodes perform a key translation process, which in essence decrypts the ingress message and then encrypts, with a new key, the egress message.
3. Can not break the standard[1] (strong) encryption algorithms.
4. Can not compromise the smart cards, payment terminals, scheme operator or CIB.

[1]Standardised in the relevant up to date statement

5. Might collude with other adversaries that steal smart cards from genuine cardholders.

In the next subsequent sections, the report describe two potential risk scenarios. The attacker capabilities identified above is taken into account in the discussion.

3.2.3 Two Potential Risk Scenarios

In this section, a discussion is carried out about two potential risk scenarios that pose a threat to the security of payment transactions. In the identification of these attacks the operating environment of the payment network, the assumptions made and the capabilities of the adversary are taken into consideration. Successful attacks that we describe below might result in compromise of the OPV process and the online transaction authorisation.

It must be noted that for the authorisation entity or CIB, it might be extremely challenging to detect the fraudulent transaction during the payment stage or even after the transaction has taken place. This is because there is a trust assumption in the payment network that numerous intermediary nodes are trusted and the adversary compromises one of these intermediaries. Therefore, it is difficult to pinpoint where the payment data was compromised and how. Furthermore, once the fraudulent transaction appears in the bank statements the genuine cardholder will deny making any payment and claim a refund from the CIB. There is a considerable financial loss associated with this type of fraud for the CIB.

Correct PIN in OPV Message

The first risk scenario is outlined in this section. In the operating environment of the payment architecture, we outlined that the adversary has complete access to the compromised intermediary node and can observe all the transactional messages that pass through the compromised node. The steps involved in the compromise of the payment transaction in the first risk scenario is listed below.

1. First, the adversary observes the communication passing through the compromised intermediary node and builds a database of PIN numbers. This database contains the Primary Account Number (PAN) and associated PIN.

2. A malicious accomplice $\mathcal{M}$ of the adversary steals a smart card. The adversary matches the cards PAN with the database. If a match is found then the adversary and the accomplice know the associated PIN. Before the card gets blocked by the

CIB, the accomplice presents the card to a (genuine) payment terminal, enters the correct PIN and can perform online or offline payment transactions.

The risk associated with the compromise is exacerbated if the relevant compromised details are cloned into a magnetic strip card and then used with the compromised correct PIN. The cloned magmatic strip card are then used in countries that have not yet fully migrated to the Chip & PIN payment scheme. This has led to a significant increase in card payment fraud that has been carried out abroad on payment cards that have been issued in countries already using the Chip & PIN scheme [93]. Most of the countries that have not migrated to the Chip & PIN scheme are still using magnetic stripe payment. Mostly all magnetic strip payment transactions are approved online and this includes the OPV as well. As we have explained in this risk scenario, the adversary is able to compromise the correct PIN and related data. With the correct PIN already being compromised, an adversary is able to successfully prove knowledge of the cardholder's PIN to the authorisation entity during a fraudulent transaction carried out by a cloned magnetic stripe card.

OPV Response Message

In the second risk scenario, a payment transaction may potentially be compromised in the following manner:

1. A malicious accomplice $\mathcal{M}$ of the adversary steals a smart card and then presents this card to a payment terminal that uses an intermediary node that is under total control of the adversary.

2. $\mathcal{M}$ selects the payment terminal in a manner that will opt in for the OPV process.

3. The payment terminal requests the cardholder ($\mathcal{M}$) to enter his PIN. $\mathcal{M}$ enters any random sequence at the payment terminal. The terminal then encrypts this entered PIN and sends it to the authorisation entity over the network.

4. The adversary captures this message. Whether it allows the message to go forward to the authorisation entity or not makes no difference. Authorisation entities (e.g. scheme operators and CIBs) do not link the OPV process with the online transaction authorisation (i.e. ARQC [16]). The adversary then replays a successful OPV verification response message generated by the CIB (observed in previous genuine transactions) back to the terminal. The adversary has observed this successful message in previous runs of the OPV process and can just replay it to the payment terminal.

5. The payment terminal will receive a successful PIN verification message and proceeds to request the smart card to generate an online transaction authorisation message (i.e. ARQC) in the 1^{st} GENERATE AC command [14]. The card generated ARQC is then sent to the authorisation entity. The authorisation entity verifies the ARQC and does a credit check on the users account. If satisfied, the authorisation entity generates an Authorisation Response Code (ARC). The ARC is then XORed with the ARQC and then enciphered using the session key shared with the card to construct an Authorisation Response Cryptogram (ARPC) to approve the transaction. The ARPC is then sent back to the CTPOS. The ARPC could also be in the form of a Message Authentication Code (MAC).

6. The CTPOS forwards the ARPC to the card and requests an outcome in the 2^{nd} GENERATE AC command [14]. The card after verifying the valid ARPC, generates a Transaction Certificate (TC) and sends it to the CTPOS.

7. CTPOS now accepts the card transaction and either sends the TC straight for payment processing or stores it for payment processing at a later time. The attack is possible due to lack of strong binding between the OPV and the ARQC.

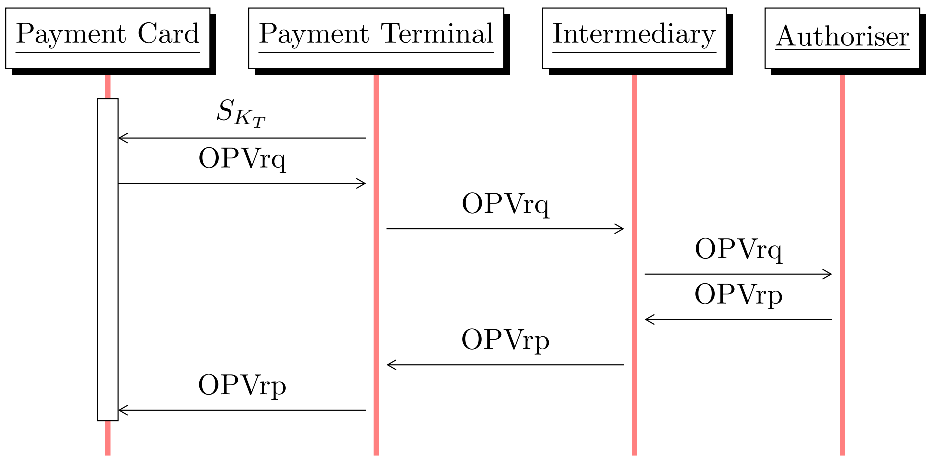
In this section, we outlined two main risk scenarios and walked through each potential payment transaction compromise. The discussed risk scenarios defines the threat model, in which our proposed solutions are based. The solutions that address the security concerns discussed above are presented in the next section.

3.3 Proposed Solutions

In this section, we present three proposed solutions that address the aforementioned risk scenarios that were discussed in Section 3.2. Overall the solutions are proposed to guarantee the end-to-end security of OPV process between the payment card and the issuing bank.

When constructing the solutions, we have taken into consideration the overhead it adds on the intermediaries in a practical term. Therefore, the proposed solutions introduce minimal or no changes to the operational/system architecture of the intermediaries involved in the current EMV payment scheme between the payment terminal and the SO/CIB. Another reason for this consideration is that the CIB has no control over the intermediaries and enforcing a system/process update on the intermediaries is a challenging task.

To meet our objective, the changes require to apply the proposed improvements are only made to the payment cards, the CIB and the CTPOS devices. The CIB has



Terminal Generated Session Key (S_{K_T}), Online PIN Verification Request (OPVrq), Online PIN Verification Response (OPVrp)

Figure 3.2: Generic OPV Protocol Sequence Diagram

complete ownership of the payment cards that are being issued to customers and their back-end authorisation systems. This means any changes to the payment cards and the back-end systems are within the capability of the CIB. Furthermore, it is considered possible to apply the security improvements to the CTPOS devices (this may be carried out by a software/hardware update) by requesting an update via the payment scheme.

We first introduce a generic OPV model that lays the foundation on which our three proposed solutions are based. The generic model also makes it easier to explain and describe the three proposed solutions more clearly. The protocol message flow of the generic OPV model is illustrated in the protocol sequence diagram in Figure 3.2.

In the generic protocol, first the CTPOS sends the payment card a session key S_{K_T} generated by the CTPOS and the PIN entered by the cardholder. The PIN is either sent in plaintext or in enciphered format.

In the event of the PIN being sent to the card in the enciphered format, the CTPOS can choose one of the following two methods depending on the scenario. They are: 1) the CTPOS could use the card's public key recovered from the card's public key certificate, 2) if the payment card contains a dedicated key pair for PIN encipherment, the CTPOS may use a card owned PIN encipherment public key.

However, it must be noted that if the PIN is sent enciphered in a PIN block, the purpose of this message is for the card to retrieve the PIN, but not to respond to a VERIFY command carried out in offline PIN verification.

After successfully receiving the message from the CTPOS, the payment card constructs a PIN block which contains the PIN and details of the corresponding account number according to ISO-9564-1 & ISO-9564-2 [17, 9]. Table 3.1 lists the data included in the OPV PIN Block.

The Unpredictable Numbers (UN) mentioned in this study have the same properties as defined in the EMV specification [14]. A symmetric key shared with the authorisation entity is used for the encipherment that is carried out inside the payment card. The authorisation entity during a payment transaction could either be the SO or the CIB. The encipherment is given a term called the OPV Encipherment PIN Block and has the notation $e\{PB\}$.

In order to provide confidentiality to the PIN related data both OPV PIN Block (PB) and the OPV PIN Result Block (PRB) are in the encrypted format. In the encipherment of the PB and the PRB using a symmetric encryption algorithm, three separate symmetric encipherment methods are used. The three encipherment methods are listed and details given below;

1. **Basic Encryption:** Advanced Encryption Standard (AES) [115] as the symmetric encryption algorithm with Cipher Feedback Mode (CFB) as the mode of operation.

2. **Encrypt-then-MAC:** AES as the symmetric encryption algorithm with CFB as the mode of operation and a key based Message Authentication Code (MAC) computed using SHA256 [149] to provide integrity.

3. **Authenticated Encryption:** AES as the symmetric encryption algorithm and Galois/Counter Mode (GCM) [135, 90] as the mode of operation is used as a combined single operation to provide authenticated encryption.

In this section, three separate messages that detail each of the symmetric encryption process are not given. Instead, it must be noted that the symmetric encipherment of the OPV PIN Block with the notation "$e\{PB\}$" represents any one of the three cryptographic processes listed above depending on which method is selected. The same symmetric encipherment algorithm used to create the $e\{PB\}$ is selected by the CIB to encipher the PRB to create the enciphered OPV PIN Result Block "$e\{PRB\}$".

For the symmetric encipherment of the OPV PIN Block in the generic protocol, we use the basic encryption mode with three blocks of 16 bytes each with a total length (L) of 48 bytes. The first block includes, 1 byte data header and 15 bytes of the random number (UN) generated by the smart card. The second block includes, the remaining 1 byte of the random number, 8 byte PIN Block and 7 bytes of the $CTPOS$ generated session key. The third block includes, remaining 9 bytes of the $CTPOS$ generated session key and 7 bytes of random padding. In our practical implementation of this, we used 128bit - AES [115] with CFB as the mode of operation.

Table 3.1: OPV PIN Block (PB) in Segmented Authorisation

Data Header	:	1 Byte.
Card Unpredictable Number (CUN)	:	16 Bytes.
PIN Block	:	8 Bytes.
CTPOS Session Key (S_{K_T})	:	16 Bytes.
Random Padding	:	7 Bytes.

Following this an Online PIN Verification Request (OPVrq) message addressed to the authorisation entity is constructed by the payment card. The message includes a concatenation of the Primary Account Number (PAN) which is the long number embossed on the payment card, and the encipherment of the OPV PIN Block $e\{PB\}$. A session key shared between the payment card and the CIB is used to encipher the $e\{PB\}$.

$$OPVrq = PAN||e\{PB\}$$

The constructed OPVrq is then sent to the CTPOS. Once the message is received by the CTPOS, it encrypts the PIN-related data by following the existing EMV process. For this the CTPOS uses the key it shares with the next entity on the communication path. This entity could be the acquiring bank or an intermediary in the EMV payment architecture. In a similar manner the OPVrq is then forwarded from entity to entity until the authoriser is reached.

Once the request is received by the authorisation entity, it deciphers the $e\{PB\}$ by retrieving the shared session key it holds in its records. The authorisation entity then verifies the recovered PIN with the PIN it holds for the cardholder in its record. If the PIN validates successfully the outcome is included in an OPV PIN Result Block (PRB). The data included in the PRB is listed in Table 3.2.

Table 3.2: OPV PIN Result Block (PRB) in Segmented Authorisation

Data Header	:	1 Byte.
Cardholder Verification Result(CVR)	:	5 Bytes.
Authoriser Unpredictable Number	:	16 Bytes.
Card Unpredictable Number	:	16 Bytes.
Random Padding	:	10 Bytes.

The Authoriser then constructs the OPVrp. This message is in enciphered format. We use the basic encryption mode with three blocks of 16 bytes each for the encipherment (128bit AES). Therefore, the last block was included with 10 bytes of random padding. It is the encipherment of the PRB using the session key S_{K_T} of the CTPOS. The notation below is used to show the symmetric encryption of PRB using the key S_{K_T}.

$$OPVrp = e\{PRB\}$$

The constructed OPVrp is then sent to the same communication path, which may go through the same acquirer and intermediaries in the current EMV architecture. The OPVrp is pushed until it is reached by the CTPOS. The message is then deciphered by the CTPOS to obtain the PIN verification result.

The CTPOS transfers this message to the payment card once the CTPOS is satisfied with the OPVrp. Based on our threat model, the payment card, CTPOS and scheme operator/CIB are assumed to be trusted entities. EMV transaction authorisation continues following the OPV process described above.

3.3.1 Card based Solutions

In this section, we discuss the proposed solutions that are mainly based on the payment card. In the card-based solution, the encipherment of the PB happens inside the payment card before it is forwarded to the CIB (Authoriser in our generic model) to be verified. The card-based solutions are further sub-categorised, depending on whether a symmetric or an asymmetric cryptographic key is used to encipher the PB.

Card uses an online-PIN encipherment symmetric key of the CIB

The solution described here introduces an online-PIN encipherment symmetric key K_{OPV} that the payment card shares with the CIB. From the K_{OPV}, a session key $K_{S_{OPV}}$ is derived using a key derivation function similar to the one specified in EMV specification [14, see: p127 - p131] and also discussed in [133]. It is assumed that the session key derivation between the card and the CIB is synchronised.

During a payment carried out using this solution, first the card is inserted to the CTPOS. Then the cardholder entered PIN is sent to the payment card. The PIN is sent either in clear or in encrypted format. The CTPOS also sends the session key S_{K_T}. The OPV process in this solution follows exactly the same steps as described in the generic OPV model above. Considering the ownership of the payment card by the CIB, the necessary changes are under the control of the CIB.

Card uses an online-PIN encipherment public key of the CIB

Previously we described a solution that is based on a symmetric key. The solution described here introduces an online-PIN encipherment public key $P_{OPV_{CIB}}$ of the CIB. It must be noted that this is not the CIB's public key that is recovered from the CIB's public key certificate residing in the card during EMV transactions. The $P_{OPV_{CIB}}$ in this solution refers to a specific online-PIN encipherment public key introduced in our construction.

This particular cryptographic key is stored in the payment card by the CIB during card personalisation. It is in the format of a public key certificate that has been signed by the CIB.

During a payment transaction in this solution, there are similar steps as in the generic model. First, the CTPOS sends a session key S_{K_T} and the PIN entered by the cardholder either in plain text or in enciphered format to the card. The card then constructs the PB-1. Table 3.3 lists the data included in the PB-1. In this occasion, PB-1 is enciphered using the public key $P_{OPV_{CIB}}$ to generate the enciphered OVP PIN Block that has the notation $z\{PB\text{-}1\}$. For the implementation of the public key-based solutions, we selected 1048bit RSA (Rivest, Shamir and Adleman) [176] with random padding generated by the selected PRNG. The included random padding was based on the Public-Key Cryptography Standards (PKCS #1 v1.5) for RSA [116]. The required random padding of non-zero randomly generated bytes was calculated as follows: 256 bytes (the public key length) deducted by 44 bytes (the message length of 41 bytes and 3 bytes for separating the padding).

Table 3.3: OPV PIN Block -1 (PB-1) in Segmented Authorisation

Data Header	:	1 Byte.
PIN Block	:	8 Bytes.
Card Unpredictable Number (CUN)	:	16 Bytes.
CTPOS Session Key (S_{K_T})	:	16 Bytes.
Random Padding	:	212 Bytes.

Following this, the OPVrq is constructed by the payment card. This message also includes the PAN and the public key encipherment of PB-1.

$$OPVrq = PAN||z\{PB-1\}$$

The constructed OPVrq is then sent to the CTPOS. The CTPOS device now follows the existing EMV process and encrypts the PIN-related data with the key it shares with the next entity on the communication path. This way the OPVrq is forwarded until the CIB is reached.

Once the message is received by the CIB, it is deciphered to obtain the PIN related data and verifies whether the PIN is correct. The outcome of the PIN verification is included in an OPV PIN Result Block (PRB-1). The content of the (PRB-1) is listed in Table 3.4

Table 3.4: OPV PIN Result Block -1 (PRB-1) in Segmented Authorisation

$Data\ Header$	:	1 Byte.
$Cardholder\ Verification\ Result(CVR)$	:	5 Bytes.
$CIB\ Unpredictable\ Number$	:	16 Bytes.
$Card\ Unpredictable\ Number$	:	16 Bytes.
$Random\ Padding$	:	10 Bytes.

Following this, the OPVrp is constructed by the CIB. This is the encipherment of the PRB-1 using the session key S_{K_T} of CTPOS.

$$OPVrp = e\{PRB - 1\}$$

The constructed OPVrp is sent back via the same communication path until the CTPOS is reached. Once the message reaches the CTPOS, the CTPOS deciphers the $e\{PRB\text{-}1\}$ to obtain the CVR. The CVR is then verified and if the verification is successful, the CVR is passed to the payment card. EMV transaction authorisation continues following the OPV process we described above.

3.3.2 Terminal based Solutions

In this section we present the terminal based solution that improves the security of the OPV process. In this solution, before the PIN block is sent to the CIB, it is enciphered at the CTPOS. This method is different to the previous two solutions that we discussed as the PIN entered by the cardholder is not sent to the card but instead enciphered in the CTPOS.

Terminal uses an online-PIN encipherment public key of the CIB

In order to carry out the OPV PIN Block encipherment in the CTPOS, this solution introduces an online-PIN encipherment public key $P_{OPV_{CIB}}$ of the CIB. A public key certificate is used to store the $P_{OPV_{CIB}}$ as in the previous Section 3.3.1. The certificate is given to the CTPOS during a payment transaction.

During a transaction, the card provides; the CIB's public key certificate (also known as the card key) signed by a Certification Authority (CA) and the CIB's online-PIN encipherment public key certificate signed by the CIB. To verify the authenticity of the CIB's public key to have been signed by the CA, the CTPOS uses the CA's public verification key. The CTPOS used the CIB's public verification key to verify that the $P_{OPV_{CIB}}$ recovered by the certificate was signed by the CIB.

The CTPOS then uses the $P_{OPV_{CIB}}$ to create the public key encipherment of PB-2 that has the notation $z\{PB\text{-}2\}$. Table 3.5 includes the data fields included in the OPV PIN Block PB-2.

Table 3.5: OPV PIN Block -2 (PB-2) in Segmented Authorisation

$Data\ Header$	:	1 Byte.
$PIN\ Block$	:	8 Bytes.
$CTPOS\ Unpredictable\ Number$	:	16 Bytes.
$CTPOS\ Session\ Key\ (S_{K_T})$	:	16 Bytes.
$Random\ Padding$	:	212 Bytes.

Following this the OPVrq is constructed by the CTPOS. The message includes the PAN and the public key encipherment of the OPV PIN Block as detailed below.

$$OPVrq = PAN||z\{PB-2\}$$

Following the communication path towards the CIB, the OPVrq is then sent from one entity to the other. Once the OPVrq is received by the CIB, the $z\{PB\text{-}2\}$ is deciphered and the PIN entered by the cardholder is verified whether to be correct or not. The outcome of this verification is included in a CVR. Afterwards, the CIB constructs the OPV PIN Result Block (PRB-2) as shown in Table 3.6 .

Table 3.6: OPV PIN Result Block -2 (PRB-2) in Segmented Authorisation

$Data\ Header$	:	1 Byte.
$Cardholder\ Verification\ Result(CVR)$	:	5 Bytes.
$CIB\ Unpredictable\ Number$	:	16 Bytes.
$CTPOS\ Unpredictable\ Number$	:	16 Bytes.
$Random\ Padding$	:	10 Bytes.

The CIB uses the session key S_{K_T} of CTPOS to encipher the PIN Result Block and includes this in the OPVrp as shown below.

$$OPVrp = e\{PRB - 2\}$$

The CIB uses the same communication path to send the OPVrp back to the CTPOS. Once the message reaches the CTPOS, the PIN result block is deciphered and the CVR is retrieved for verification. Once satisfied, the CTPOS will generate other commands if the transaction proceeds to transaction authorisation.

3.3.3 Binding of OPV and Transaction Authorisation

In the current EMV process there seems to be no direct linkage between the OPV process and the online transaction authorisation for a given EMV transaction. As the two verifications are carried out separately in two different instances, this raises a security concern. This leaves a space for replay attacks in which a harvested OPV Response Message could be replayed or injected by the compromised intermediary during an EMV transaction.

Both of our card-based and terminal-based proposals discussed in sections 3.3.1, 3.3.2 help to eliminate the aforementioned attacks by making a minor change to the current EMV transaction authorisation message. In response to the GENERATE AC command issued by the CTPOS in an EMV online transaction authorisation process, the payment card generates an Authorisation Request Cryptogram (ARQC) [14, 15].

Here the payment card can include the *Authoriser Unpredictable Number* inside the ARQC, which is a symmetric encipherment using the shared key between the card and the Authoriser. The Authoriser Unpredictable Number is sent back to the CTPOS in the OPV PIN Result Block (PRB) as discussed in our generic OPV model in Section 3.3 and shown in Table 3.2.

Once the ARQC is received, the authoriser deciphers it. The Authoriser can use the *Authoriser Unpredictable Number* that it keeps a record of to link the previously verified OPV to the received transaction authorisation request. This gives assurance to the Authoriser that this is a genuine and timely transaction. The Authoriser may also include a combined verification result inside the Authorisation Response Cryptogram (ARPC) when the ARPC is sent back to the card.

3.3.4 Our Experience Related to EMV Specifications

The proposed solutions improves the security of the OPV process within the threat model identified in Section 3.2.3. Improving the OPV would mean that the strong trust assumptions that are placed on the intermediary nodes can now be relaxed. The solutions are categorised into card-based and terminal-based solutions depending on which entity the PIN block encipherment occurs in during OPV.

We have carried out a thorough research and studied the EMV specifications to learn and understand PIN block construction. The EMV specifications do not provide any details in regard to the PIB block construction and encipherment of the OPV [13, 14, 15, 16]. However, one of the EMV specifications details the process of PIN block construction and encipherment that happens at the CTPOS during an offline PIN verification process which is different to the OPV [14]. In an offline PIN verification, the CTPOS sends the enciphered PIN to the card to be verified [14].

Furthermore, in our study, standards such as ISO-9564-1 & ISO-9564-1 and similar guidelines given in [195, 17, 9] were referred. These documents make recommendations on how PINs and associated account information need to be protected during transmission from one system to another. Since there is no publicly available standard on how PIN block construction and encipherment should be carried out in OPV, we have made reasonable assumptions in our PIN block construction and encipherment.

3.4 Analysis

In this section, the proposed solutions presented in the previous section are evaluated for their security and performance. Firstly, the security of the proposed solutions is evaluated while taking into consideration the attacker's capability. Afterwards, performance measurements are taken to show the potential penalties for the existing process if they are adopted.

3.4.1 Security Analysis

Before we begin are analysis the capabilities of the adversary outlined in section 3.2.2 is revisited. According to this, a malicious user (adversary) has the capability of compromising an intermediary entity. However, the adversary cannot compromise the smart cards, payment terminals, scheme operator, or CIB. This limitation is imposed because if an adversary can successfully compromise any of these entities then almost no protection mechanism would be strong enough to protect against attacks on the OPV process. The rationale behind this is described below:

1. During an offline PIN-based transaction, the payment card uses the copy of the cardholder PIN it securely stores in its tamper resistant chip to compare with the value entered by the cardholder. If an adversary can break the tamper resistant smart card to access the PIN, then he could potentially access any other infor-

mation on it, thus rendering any countermeasure, including ours, redundant. In such scenario, the attacker is able to clone a completely new card with the details and perform a transaction with the correct PIN.

2. If an attacker has the capability of successfully compromising a payment terminal, then the attacker can access authorised PIN at all instances a payment card is being used at that particular compromised terminal. However, in this scenario, the adversary will only capture the PINs of the cards used on a single payment terminal.

3. An authorisation entity (e.g. scheme operator and CIB), similar to the payment cards, has to store copies of authorised PINs for verification purposes. If an adversary compromises the authorisation entity then it is challenging to protect the PINs and the OPV process (even with our proposal).

In our proposed solutions, secure communication to the authorising entity is provided by using the OPVrq. In particularly, the payment card issued by the respective CIB either encrypts the message itself or gives the encryption (public) key to the payment terminal to be used for encryption.

After receiving the OPVrq by the CIB, it is deciphered and the PIN is verified. The response (OPVrp) which indicates whether the PIN verification was successful or not is sent back to the payment terminal. The payment terminal generates the session key that encrypts the OPVrp, which is included in the OPVrq by either the smart card or the terminal. The OPVrp also guarantees message freshness. Depending on the selected proposed solution, this is achieved by either the payment card or the terminal generating an unpredictable number and including this in the OPVrp.

An adversary observing these messages can store them for the purpose of replaying them at some later stage. For the solution based on the symmetric key and encryption performed by the smart card, a replay of the OPVrq will be easily detected as the session key used for encrypting (and successful decryption) this message would have expired. Furthermore, the adversary cannot see the PIN in plaintext as he does not have the capability of breaking a strong cryptographic algorithm.

However, for solutions based on an asymmetric cryptosystem (e.g. using public key of the CIB) the smart card or the payment terminal generates a session (symmetric) key. This key is later used by the CIB to encrypt the OPVrp. If the adversary replays the OPVrq message, which is enciphered using the public key of the CIB, the session key part of the replayed message would be different. As a result, when a OPVrp is generated by the CIB, the payment terminal may not be able to decrypt this OPVrp message properly. This would avoid a successful replay attempt of the OPVrq message.

In section 3.3.3, we explained how our proposed solution binds the OPV process with the online transaction authorisation process. This is achieved by including the authorising entity generated *Authoriser Unpredictable Number* sent back in the PIN Result Block in the online transaction authorisation. The authorising entity can use the unpredictable number to link a particular OPV to its associated online transaction authorisation. This provides a countermeasure against an adversary taking advantage of the lack of binding between the OPV process and the online transaction authorisation.

Consider a potential scenario in which an adversary creates a "Yes" card [146]. A "Yes" card is a fake payment card that replies to the terminal with a PIN verification successful message regardless to whether the entered PIN is correct or not. In this scenario, the OPV process has to execute the first two proposed solutions (based on the smart card).

When a malicious user enters a PIN on the payment terminal, it is sent to the "Yes" card. Following this, the "Yes" card generates an OPVrq message. The intention is that when the authorisation entity tries to verify the OPVrq and fails, it will send a PIN verification decline result back in the OPVrp. However, this response goes bank to the "Yes" card which simply discards this message. Instead the "Yes" card sends the payment terminal a message that indicates to the payment terminal that the PIN verification was successful.

The success of this attack is dependent on the OPVrq completely being isolated from the payment terminal. We remove this isolation in our proposed solution by using a symmetric key generated by the payment terminal. After generating the symmetric key, it is communicated to the authorisation entity in the OPVrq message. The symmetric key is used by the authorisation entity to encipher the OPVrp. Therefore, once the OPVrp is received at the payment terminal, it can also decrypt the message and verify whether the PIN was verified or not. Hence, this potential scenario cannot circumvent the protection provided by our proposals.

In our analysis, we consider another concern that could raise some security concerns. That is the use of a standard Initialisation Vector (IV) [138] for the symmetric cryptosystem. The reason why we discuss this is because of the potential patterns in the ciphertext might reveal some information regarding the PIN. To avoid this (even when the IV is a predefined value), in the OPVrq message, we append a random number generated by the smart card to the data header and then append the PIN value. This way, the first 16 byte (plaintext) block to be used by the symmetric algorithm (i.e. AES [115]) will have 15 random bytes and the first byte of the second block will also be random (the 16 byte random number is spread over the first two plaintext blocks). This randomness in the plaintext of the first block avoids any patterns being detected in the second block, which contains the PIN. Furthermore, the session keys are unique per transaction for the symmetric key-based solutions. Which means that the same key is not used in multiple transactions. The encryption makes it difficult for an adversary to gain any additional information from the OPVrq or OPVrp message about the PIN or the associated decision.

3.4.2 Practical Implementation

In this section, we describe our implementation of the proposed solutions. The main aim of implementing the proposed solutions was to identify and measure potential performance penalties the existing OPV process has to bear if our proposed improvements were to be applied.

To implement our solutions, we used two 32bit Java Cards [12] connected with a Microsoft Windows 7 machine running on 2.3GHz, 2GB RAM as our test bed. The machine was used to represent the CTPOS and CIB (for OPVrp). For the symmetric key based solution, we selected Advanced Encryption Standard (AES) [115] as the encryption algorithm with a 128bit key.

Table 3.7: Performance Penalties associated with Each Solution

Proposed Solutions	**Basic Encryption**		**Encrypt-then-MAC**		**Authenticated Encryption**	
	Java Card 1	**Java Card 2**	**Java Card 1**	**Java Card 2**	**Java Card 1**	**Java Card 2**
Symmetric Key Card	64ms	86ms	138ms	156ms	122ms	136ms
Asymmetric Key Card	138ms	159ms	196ms	218ms	-	-
Asymmetric Key Payment Terminal[2]	34ms		48ms		-	
CIB OPVrp (footnote 7)	16ms		28ms		22ms	

We used the Cipher Feedback Mode (CFB) as the mode of operation for both basic encryption and Encrypt-then-MAC methods. Whereas, we opted for the Galois/Counter Mode (GCM) as the authenticated encryption method [135, 90]. For the generation of the random number [35] we selected the HMAC-based Pseudorandom Number Generator (PRNG)[3] .

For the implementation of the public key-based solutions, we selected 1048bit RSA (Rivest, Shamir and Adleman) [176] with random padding generated by the selected PRNG. The same PRNG was used to generate the session keys used in this solution.

Table 3.7 lists the performance penalties incurred by our proposed solutions. All measurements are given in milliseconds (ms). A point to note is that there is no authenticated encryption mode (similar to GCM) for asymmetric cryptosystems (i.e. RSA). Therefore, in Table 3.7 performance measurement for this is not included.

Our implementation does not emulate the complete EVM specifications. It only

[3]PRNGs based on different cryptographic algorithms can give different performances; a detailed discussion of this can be found in [35]

implements the proposed improvements to the OPVrq and OPVrp processes. The performance measurement should be taken as an additional execution cost that the EVM process has to bear to implement the proposed solutions in this chapter.

3.4.3 Mechanical Formal Analysis

In this section, the proposed improvements are subject to a mechanical formal analysis. The analysis is based on the CasperFDR tool. The CasperFDR mechanical analysis framework can be used to test the soundness of a security protocol under a set of defined security properties. In this approach, the Casper compiler [129] takes a high-level description of the protocol, together with its security requirements. It then translates the description into the process algebra of Communicating Sequential Processes (CSP) [105]. The CSP description of the protocol can be machine-verified using the Failures-Divergence Refinement (FDR) model checker [172]. The intruder's capability modelled in the Casper script (appendices A.1.1 and A.1.2) for the proposed protocol is:

1. Intruder can masquerade any entity in the network
2. Intruders can read the messages transmitted in the network
3. Intruder cannot influence the internal process of an entity in the network

The security specification for which the CasperFDR evaluates the network is as shown below. The listed specifications are defined in the #Specification section of appendices A.1.1 and A.1.2:

1. The protocol run is fresh and both applications were alive,
2. The key used for encryption/decryption in the symmetric system and the private key used for decryption in the asymmetric system, is not revealed to the adversary,
3. Long terms keys of communicating entities are not compromised, and

After successfully evaluation the proposed protocol, the CasperFDR tool did not find any feasible attacks and weaknesses related to the protocol.

3.5 Summary

In this section, we conclude our discussion and summarise the key contributions of this chapter.

At the beginning of this chapter, the EMV OPV process, current payment architecture and the online transaction authorisation process were discussed. We then identified and described how certain aspects of the payment architecture and its associated deployment methods open up a potential route for an adversary to compromise payment transactions for fraudulent financial gains. Afterwards, assumptions related to the payment network's operating environment, the capabilities of an adversary and potential attack scenarios were outlined.

Subsequently, we proposed three potential ways to enhance the OPV process and a proposal of how to bind it to the online transaction authorisation. The proposed solutions were then analysed with a discussion on their security in the context of the adversary's capabilities. We also provided the execution measurements for our proposed modifications; this showed the potential performance penalty incurred by our proposals.

Furthermore, proposed modifications were then subjected to the mechnical formal anlysis using the CasperFDR tool. The concerns raised by this chapter are considered to be valid as the OPV and online transaction authorisation is considered the highest level of trust in the card-based payment mechanism. It can differ based on laws/regulations or the relationship between the cardholder and CIB, but if the correct PIN is used in an OPV and online transaction authorisation then the liability of the payment is either with the cardholder or the CIB. If attacks can successfully occur at this level they could potentially cause substantial reputation damage to the overall card-based payment scheme, along with causing financial loss to the cardholder/CIB.

Furthermore, such attacks could make it difficult to detect whether an OPV-based transaction was actually made by the cardholder or the adversary, as the compromise of the intermediary nodes might not be detected in time. Therefore, we consider this to be a concern and suggest that a mandating rollout of an OPV process in a geographical region should take into consideration these concerns and our potential solutions.

Chapter 4

Improving OPV Security in Unified Authorisation Method

Contents

In this Chapter, we extend our work on enhancing the EMV OPV process deployed in the Unified Authorisation method in the current EMV architecture. We then identify potential attack scenarios that an attacker can use to gain access to the Personal Identification Number (PIN) data by compromising the OPV process. Addressing these concerns, we propose improvements that enhance the security of the OPV process in the Unified Authorisation method.

4.1 Introduction

The EMV OPV process can be deployed in two different methods. When OPV is carried out in a separate message from the online transaction authorisation, we named that deployment method as Segmented Authorisation. In contrast to this, when OPV is carried out in the same message together with the online transaction authorisation we named that deployment method as Unified Authorisation. In Section 3.1.1, we differentiated the two OPV deployment methods that could be used in the current EMV architecture to carry out the OPV process.

The EMV OPV process we discussed in chapter 3 was the Segmented Authorisation method. In this chapter, we extend our work carried out to enhance the EMV OPV process in the Unified Authorisation method. The transaction steps in the segmented authorisation method were discussed in Section 3.1.1 and in the unified authorisation method were discussed in Section 3.1.1.

In Section 2.1.1, we first gave a detailed introduction to EMV payment scheme and its architecture. We then introduced the EMV OPV process and showed how OPV is carried out in a generic setting in Section 2.1.2. More emphasis is given on the Unified Authorisation method as it is the main focus of this chapter. The operating environment of the current EMV payment architecture and assumptions were discussed in Section 3.2.1. A representation of the operating environment was illustrated in Figure 3.1.

In the next section, we extend our discussion on the Unified Authorisation method and investigate whether there are any security weaknesses or potential risk scenarios.

4.2 Potential Concerns

This section is more focused in identifying security weaknesses and potential attack scenarios in the Unified Authorisation method. We begin the discussion by outlining trust assumptions associated with the operating environment discussed in Section 3.2.1. Afterwards, we detail the attacker's capability in the current operating environment. Subsequently, we examine two potential risk scenarios associated with the Unified Authorisation OPV deployment method.

The current architecture shown in Figure 3.1, has placed an indelible trust assumption on the intermediary entities that engage in key translation during OPV and transaction authorisation processes. Most of these intermediaries are bound by contracts with either the acquirer or SO/CIB. Part of the contract may include information related to liability obligations and security requirements. Yet it is questionable whether this is sufficient to let intermediaries handle sensitive data related to the cards and cardholders.

The CTPOS devices are deployed by Payment Terminal Operators (PTOs) as discussed in our operating environment in Section 3.2.1. It is not practically feasible for an issuer to get in contact with all the PTOs globally, in order to share a secret cryptographic key to make communication secure between the CTPOS and the issuer. We also understood that given the number of different merchants' acquirers, subcontractors, third parties and the number of CTPOS devices, this would be logistically impractical. After all, one of the objectives of introducing EMV, was to achieve the interoperability between different entities without prior business relationships.

In this work, the smart cards, payment terminals, scheme operators and CIB are considered to be secure and trusted. The rationale for this consideration was discussed in Section 3.4.1. However the intermediaries are considered to have the potential to be compromised. This assumption is based on recent reports and incidents where banking sector services were successfully infiltrated by adversaries [45, 117, 118, 54, 110, 125]. Therefore, the assumption can be considered reasonable. Taking this operating environment and our assumptions regarding the intermediary nodes and other entities, we expand the discussion to the capabilities of our adversary in the next section.

4.2.1 Two Potential Risk Scenarios

As the operating environment and assumptions are similar to the one discussed in Chapter 3, the attacker's capabilities that we identify here are the same as discussed in Section 3.2.2. Based on the payment-networks operating environment, our assumptions and the adversary's capabilities, two potential risk scenarios that pose a threat to the OPV in the Unified Authorisation method are discussed here.

In Section 3.2.3, we identified two risk scenarios and discussed how the risks can compromise the security of OPV in the Segmented Authorisation method. From these two risk scenarios, only the first risk scenario (Correct PIN in OPV Message) discussed in Section 3.2.3 is valid in the Unified Authorisation method. The second risk scenario (OPV Response Message) discussed in Section 3.2.3 is not longer valid in the Unified Authorisation method. The main reason for this is that the OPV and the online transaction authorisation are being carried out together in the Unified Authorisation method. For further explanation regarding why the second risk scenario is not valid in the Unified Authorisation method, please refer to Section 4.4.1. However, a new risk scenario has emerged that can potentially compromise the OPV process in the Unified Authorisation method. The risk scenario is explained below.

PIN Block Replay in OPV Request

Discussed below is another potential attack scenario that pose a threat to the security of OPV. The risk scenario that we outline in this section shows how OPV can be compromised in the Unified Authorisation method. The compromise can be considered as an attack carried out to cause financial and reputational damage to one or more targeted financial institutions. The attack could also be used by criminals to blackmail financial organisation for ransom. In this risk scenario, the payment transaction compromise is carried out as follows:

1. In this attack, the adversary keeps a record of all the PAN and associated PIN blocks passing through the compromised intermediary. The adversary in addition to this, keeps a record of all the merchants by mapping the merchant IDs that are associated to the transactions that pass through the compromised intermediary.

2. The adversary may threaten the financial institution for ransom or post a message on online forums advertising that lost/stolen cards from the advertised geographical locations are more likely to be approved during transactions even by entering a random sequence for the PIN at a number of listed merchants.

3. Whenever an online authorisation message in the Unified Authorisation Method is received by the adversary at the compromised intermediary, it carries out a database string search to retrieve the corresponding PIN block of the PAN. If the PIN block is already in the adversary's database, then the adversary removes the existing PIN block part of the online authorisation message and replaces it with the correct PIN block retrieved from the database.

4. This way the adversary replays the previously known correct PIN blocks by replacing the PIN block part of the online authorisation message but leaves the ARQC part of the message unchanged. The message is then forwarded towards the CIB.

5. The CIB after receiving the online authorisation message, which includes both the OPV and the online transaction authorisation, first verifies the PIN and if correct proceeds to online transaction authorisation. The CIB verifies the ARQC and does a credit check on the user's account. If satisfied, the CIB generates an ARC. This is then Xored with the ARQC and enciphered using the shared session key with the card to generate the ARPC.

6. The CIB then sends the OPV result and the ARPC to the terminal in a single response message. The terminal checks the successful PIN verification result from

the CIB. Upon successful OPV result, the terminal forwards the ARPC to the card and request an outcome in the 2^{nd} GENERATE AC command.

7. The card after verifying the validity of the ARPC, generates a Transaction Certificate (TC) and sends it to the terminal. The terminal then approves the transaction.

In this section, we identified two attack scenario that pose a significant threat to the OPV process when it is deployed in the Unified Authorisation method. In the two risk scenarios, due to the attacks being performed at the compromised intermediary, it is difficult for the authorisation entity to detect a fraudulent transaction. We also identified another potential attack scenario but discussed why it cannot be carried out in the Unified Authorisation deployment method. Aiming to address the aforementioned security concerns, we present our proposed solutions in the next Section.

4.3 Proposed Solutions

In this Section, we propose three solutions that address the aforementioned security concerns and potentially guarantees end-to-end security of OPV between the payment card and the CIB. In our approach we have also taken in to consideration the operating environment of the payment architecture and attacker capability.

The proposed solutions are compatible with both OPV deployment methods. However, we have already presented the solutions for the Segmented Authorisation method in Section 3.3. Therefore, in this section we only emphasise in presenting the solutions that are relevant for the Unified Authorisation method.

When constructing the solutions for this deployment method, as before we have taken into consideration the overhead it adds on the intermediaries in practical terms. Therefore, the proposed solutions introduce minimal or no changes to the operational/system architecture of the intermediaries involved in the current EMV payment scheme between the payment terminal and the SO/CIB. To guarantee this, the required changes to apply the proposed improvements are only made to the payment cards, the CIB and the CTPOS devices.

The proposed solutions remove the need for placing strong trust assumptions on the intermediary entities. The solutions are categorised into card-based and terminal-based solutions depending on which entity the PIN block encipherment occurs in during the OPV process. For a generic construction of the OPV process and protocol diagram we would direct the reader to Section 3.3 of the previous Chapter 3.

In our construction, the EMV specifications, standards such as ISO-9564-1 & ISO-9564-2 and similar guidelines were referred [13, 14, 15, 16, 195, 17, 9]. Since there is no publicly available standard on how PIN block construction and encipherment should be carried out in OPV, we have made reasonable assumptions in our PIN block construction and encipherment.

4.3.1 Card Based Solutions

In this section, we present our solutions that are mainly based on the payment card. In the payment card based solution, the PIN Block (PB) is enciphered inside the payment card before it is forwarded to the CIB (Authoriser in our generic model) to be verified. The card-based solutions are further sub-categorised, depending on whether a symmetric or an asymmetric cryptographic key is used to encipher the PB.

Card uses an online-PIN encipherment symmetric key of the CIB

Here we present our first solution which introduces an online-PIN encipherment symmetric key K_{OPV} that is shared between the payment card and the CIB. Based on the K_{OPV}, a session key $K_{S_{OPV}}$ is derived using a key derivation function similar to the one specified in EMV specification [14, see: p127 - p131] and also discussed in [133]. It is assumed that the session key derivation between the card and the CIB is synchronised.

During a transaction, the payment card is inserted to the CTPOS and the PIN is entered. The PIN gets sent to the payment card either in plain text or in encrypted format. The CTPOS also sends the session key S_{K_T}. In the Segmented Authorisation method, the proposed solution follows exactly the same process as described in the generic OPV model in Section 3.3. Presented below is the proposed solution for the Unified Authorisation method.

In the Unified Authorisation method, the PIN is enciphered by the card using the online-PIN encipherment symmetric key in exactly the same process that is carried out in the segmented authentication method. The only difference is due to the need of transferring both the OPV and the online transaction authorisation messages in the same message towards the CIB. The construction of the proposed solution in the Unified Authorisation method is as follows:

The CTPOS first obtains the enciphered PIN block from the card, where as the card constructs it exactly the same way as before in the segmented authorisation method. The data included in the OPV PIN Block are shown in Table 4.1. The Unpredictable Numbers (UN) mentioned in this study have the same properties as defined in the EMV specification [14].

Table 4.1: OPV PIN Block (PB) in Unified Authorisation

$Data\ Header$	:	1 Byte.
$PIN\ Block$	:	8 Bytes.
$Card\ Unpredictable\ Number$ (CUN)	:	16 Bytes.
$CTPOS\ Session\ Key$ (S_{K_T})	:	16 Bytes.
$Random\ Padding$	:	10 Bytes.

However, the card generates a hash of the enciphered PIN block which has the notation $h(e\{PB\})$ and keeps the hash on its records. The stored hash is later used in the $ARQC$ construction. The CTPOS instead of sending the OPV Request to the CIB, requests an $ARQC$ from the card by issuing the 1^{st} GENERATE AC command. As opposed to the $ARQC$ generated by the card in the segmented authorisation method, in the Unified Authorisation method the $ARQC$ indicates that OPV is not carried out and the PIN is not verified. Furthermore, the card also includes the $h(e\{PB\})$. The hash of the enciphered PIN block provides an assurance that the associated PIN block has a cryptographic binding with the received $ARQC$. Due to $h(e\{PB\})$ being included in the $ARQC$ and the cryptogram being enciphered with a shared symmetric key shared between the card and the CIB, assurance to the integrity of the included hash is given. This is further discussed in Section 4.4.1. For the generation of the hash, we use SHA256 hash function [94].

The $ARQC$ is then sent to the CTPOS. To meet the mandate of having both parts in the same message, the CTPOS then constructs an Online PIN Verification and Transaction Authorisation Request (OPVTArq) message, which includes the PAN, enciphered PIN block and the $ARQC$. The message is then forwarded towards the CIB.

$$OPVTArq = PAN||e\{PB\}||ARQC$$

The CIB, upon receiving the OPVTArq, uses the PAN to retrieve the shared session key and deciphers the $e\{PB\}$. The CIB, in this instance, validates the PIN to be correct before proceeding to transaction authorisation.

[0]Measurement was taken from a desktop computer.

Table 4.2: OPV PIN Result Block (PRB) in Unified Authorisation

$Data\ Header$	:	1 Byte.
$Cardholder\ Verification\ Result(CVR)$	:	5 Bytes.
$Authoriser\ Unpredictable\ Number$	:	16 Bytes.
$Card\ Unpredictable\ Number$	:	16 Bytes.
$Random\ Padding$	:	10 Bytes.

Only if the PIN is correct, then the CIB constructs the OPV PIN Results Block (PRB), shown in Table 4.2 and encrypts it using the CTPOS's session key S_{K_T} to create the enciphered PIN results block $e\{PRB\}$, but does not send it to the CTPOS yet. Instead, it verifies whether the $ARQC$ is genuine and has been generated by the card. The fields in the $ARQC$ also indicate to the CIB that OPV has not been carried out prior to this stage. Furthermore, the CIB generates the hash of the enciphered PIN block again and compares it with the recovered $h(e\{PB\})$ from the $ARQC$. If the two hashes match, then the CIB has assurance that the received $e\{PB\}$ is associated with the $ARQC$ in the transaction. The CIB also conducts an account level credit check on the card holder, and if satisfied, constructs an $ARPC$. During the construction of the $ARPC$, the CIB also includes an unpredictable number UN_{CIB} in the $ARQC$. After this process, the CIB constructs an Online PIN Verification and Transaction Authorisation Response (OPVTArp) message, which includes the PAN, enciphered PIN result block and the $ARPC$. The message is then sent towards the CTPOS.

$$OPVTArp = PAN||e\{PRB\}||ARPC$$

The CTPOS deciphers the $e\{PRB\}$ to obtain the PIN verification results. If satisfied with the result of the PIN verification, the CTPOS forwards the message to the card and request an outcome in the 2^{nd} GENERATE AC command. The card, verifies the validity of the $ARPC$ and generates a Transaction Certificate (TC). During the construction of the TC, the card includes CUN, UN_{CTPOS} & UN_{CIB} in the TC, cryptographically binding the associated unpredictable numbers. This links the $e\{PB\}$, $e\{PRB\}$, $ARQC$ & $ARPC$ to the TC. The TC is then sent to the CTPOS, who then approves the transaction. The CTPOS either forwards the TC for payment processing straight away or keeps it in its record to be forwarded at a later time. In either way, when the TC is received, the CIB can verify that the TC has a binding to OPV and transaction authorisation processes.

Card uses an online-PIN encipherment public key of the CIB

The solution we discussed above uses a symmetric key for the PIN Block encipherment. The solution described here introduces an online-PIN encipherment public key $P_{OPV_{CIB}}$ of the CIB. It must be noted that this is not the CIB's public key that is recovered from the CIB's public key certificate residing in the card during EMV transactions. The $P_{OPV_{CIB}}$ is a specific online-PIN encipherment public key introduced in our construction which is in the format of a public key certificate that has been signed by the CIB. The solution for the Segmented Authorisation method can be found in Section 3.3.1 of the previous Chapter 3. Presented below is the proposed solution for the Unified Authorisation method.

In the unified authentication method, similar to the segmented authorisation method, the card enciphers the constructed PIN block PB-1 using $P_{OPV_{CIB}}$ to generate the enciphered PIN block that has the notation $z\{PB\text{-}1\}$. The data included in the PB-1 are shown in Table 4.3.

Table 4.3: OPV PIN Block -1 (PB-1) in Unified Authorisation

$Data\ Header$	:	1 Byte.
$PIN\ Block$	:	8 Bytes.
$Card\ Unpredictable\ Number\ (CUN)$	:	16 Bytes.
$CTPOS\ Session\ Key\ (S_{K_T})$	:	16 Bytes.
$Random\ Padding$	:	212 Bytes.

The $z\{PB\text{-}1\}$ is sent to the CTPOS, who then instead of sending an OPV request to the CIB, issues the 1^{st} GENERATE AC command to the card requesting the $ARQC$.

The card then constructs an $ARQC$, the card also includes the hash of the public key enciphered PIN block, which has the notation $h(z\{PB\text{-}1\})$. The $ARQC$ is then sent to the CTPOS. The CTPOS in possession with both the $z\{PB\text{-}1\}$ and the $ARQC$: constructs an Online PIN Verification and Transaction Authorisation Request (OPV-TArq) message, which includes the PAN, enciphered PIN block and the $ARQC$. The message is then forwarded towards the CIB.

$$OPVTArq = PAN||z\{PB-1\}||ARQC$$

The CIB, upon receiving the OPVTArq, deciphers the $z\{PB\}$ and verifies whether the PIN is correct. The outcome of this verification is constructed in an OPV PIN Result Block (PRB-1) shown in Table 4.4

Table 4.4: OPV PIN Result Block -1 (PRB-1) in Unified Authorisation

$Data\ Header$	:	1 Byte.
$Cardholder\ Verification\ Result(CVR)$	:	5 Bytes.
$CIB\ Unpredictable\ Number$	:	16 Bytes.
$Card\ Unpredictable\ Number$	:	16 Bytes.
$Random\ Padding$	:	10 Bytes.

This is then enciphered using S_{K_T} to create the enciphered PIN Result Block $e\{PRB - 1\}$. If the PIN verification was successful, the CIB proceeds to transaction authorisation by verifying the $ARQC$ and doing an account credit check. During the $ARQC$ verification, the CIB generates the hash of the public key enciphered PIN block and compares it with the hash $h(z\{PB\text{-}1\})$ recovered from the $ARQC$. If the two hashes match, the CIB can verify that the received PIN block and $ARQC$ are associated with one another. The CIB then constructs the $ARPC$. Following this, the CIB constructs an Online PIN Verification and Transaction Authorisation Response (OPVTArp) message, which includes the PAN, enciphered PIN result block and the $ARPC$. The message is then sent towards the CTPOS.

$$OPVTArp = PAN||e\{PRB - 1\}||ARPC$$

The CTPOS deciphers the $e\{PRB - 1\}$ to obtain the PIN verification result. If satisfied, the CTPOS forwards the message to the card and requests an outcome in the 2^{nd} GENERATE AC command. The card verifies the validity of the $ARPC$ and generates a TC. During the construction of the TC, the card includes CUN, UN_{CTPOS} & UN_{CIB} in the TC, cryptographically binding the associated unpredictable numbers. This links the $e\{PB\}$, $e\{PRB\}$, $ARQC$ & $ARPC$ to the TC. The TC is sent to the CTPOS, who then approves the transaction. When the CTPOS forwards the TC for payment processing, the CIB can verify that the TC has a binding to OPV and transaction authorisation processes.

4.3.2 Terminal based Solution

In this solution, we present our third solution that is based on the terminal which is also referred to as the CTPOS in this Chapter. Here the PIN Block is enciphered at the CTPOS before it is sent to the CIB for verification. A noticeable difference in this method is that the PIN entered by the cardholder is not sent to the payment card but instead enciphered by the CTPOS.

Terminal uses an online-PIN encipherment public key of the CIB

In this solution an online-PIN encipherment public key $P_{OPV_{CIB}}$ of the CIB is introduced. This cryptographic key is mainly used to carry out the OPV PIN Block encipherment at the CTPOS. The public key is stored in the payment card in a public key certificate, similar to the proposal in Section 4.3.1, and during a transaction the certificate is given to the CTPOS. The solution for the Segmented Authorisation method can be found in Section 3.3.2 of the previous Chapter 3. Presented below is the proposed solution for the Unified Authorisation method.

Our third solution under the Unified Authorisation method, does not send the card holder entered PIN to the card, instead it is enciphered by the CTPOS. First the CTPOS, includes the PIN in to a OPV PIN Block (PB-2). The OPV PIN Block PB-2 includes data mentioned in Table 4.5.

Table 4.5: OPV PIN Block -2 (PB-2) in Unified Authorisation

Data Header	:	1 Byte.
PIN Block	:	8 Bytes.
CTPOS Unpredictable Number	:	16 Bytes.
CTPOS Session Key (S_{K_T})	:	16 Bytes.
Random Padding	:	212 Bytes.

It must be noted that, the unpredictable number included in is generated by the CTPOS but not the card as in the previous two solutions. The CTPOS then uses $P_{OPV_{CIB}}$ to create the public key encipherment of PB-2 that has the notation $z\{PB$-$2\}$. As opposed to the third solution in the Segmented Authorisation method, the CIB does not send the $z\{PB$-$2\}$ to the CIB for OPV but instead request the card for an $ARQC$ by issuing the 1st GENERATE AC command.

The card as in the previous two solutions in the Unified Authorisation method, includes a field inside the $ARQC$ stating that the OPV process is not carried out and the PIN is not verified. Furthermore, the card also includes the hash of the public key enciphered PIN block $h(z(\{PB$-$2\})$ in the $ARQC$. The $ARQC$ is then sent to the CTPOS. The CTPOS now in possession with the $z\{PB$-$2\}$ and the $ARQC$ constructs an Online PIN Verification and Transaction Authorisation Request (OPVTArq) message, which includes the PAN, enciphered PIN block and the $ARQC$. The message is then forwarded towards the CIB.

$$OPVTArq = PAN||z\{PB-2\}||ARQC$$

Once the message is received, the CIB deciphers the $z\{PB$-$2\}$ and verifies the

PIN. The outcome is included in an OPV PIN Result Block (PRB-2). The CIB then constructs a OPV PIN Result Block (PRB-2) shown in Table 4.6

Table 4.6: OPV PIN Result Block -2 (PRB-2) in Unified Authorisation

$Data\ Header$	:	1 Byte.
$Cardholder\ Verification\ Result(CVR)$	:	5 Bytes.
$CIB\ Unpredictable\ Number$	:	16 Bytes.
$CTPOS\ Unpredictable\ Number$	:	16 Bytes.
$Random\ Padding$	:	10 Bytes.

This is then enciphered using S_{K_T} to create the enciphered PIN Result Block $e\{PRB-2\}$.

If the PIN verification is successful, the CIB verifies the $ARQC$ and does an account level credit check. During the $ARQC$ verification, the CIB generates the hash of the public key enciphered PIN block and compares it with the $h(z(\{PB\text{-}2\})$ recovered from the $ARQC$. The two hashes must match for the CIB to complete transaction authorisation. Once transaction authorisation is completed the CIB generates the $ARPC$, which is then included in the OPVTArp message together with the PAN and the $e\{PRB-2\}$ before sending the message to the CTPOS.

$$OPVTArp = PAN||e\{PRB-2\}||ARPC$$

The CTPOS deciphers the $e\{PRB-2\}$ to obtain the PIN verification result. If satisfied, it forwards the message to the card and requests an outcome by issuing the 2^{nd} GENERATE AC command. The card after verifying the validity of the $ARPC$, generates the TC. Similar to the previous two solutions in the unified authentication method, the card includes CUN, UN_{CTPOS} & UN_{CIB} in the TC. This links the $e\{PB\}$, $e\{PRB\}$, $ARQC$ & $ARPC$ to the TC. The TC is sent to the CTPOS, who then approves the transaction. When the CTPOS forwards the TC for payment processing, the CIB can verify that the TC has a binding to OPV and transaction authorisation processes.

4.3.3 Binding of OPV and Transaction Authorisation

As we have detailed before, in the Segmented Authorisation method there seems to be no direct linkage between the online PIN verification and the online transaction authorisation for a given EMV transaction. The two verifications are carried out separately, leaving space for replay attacks in which a harvested OPV Response Message could be replayed or injected by the compromised intermediary during an EMV transaction.

In the Unified Authorisation method, due to the mandate of both the enciphered PIN block and the *ARQC* being received by the CIB at the same time makes the attack discussed in Section 3.2.3 invalid for the Unified Authorisation method. However, this leaves an opportunity for adversaries to carry out other alternative attacks, such as the attack scenario we identified in Section 4.2.1.

In contrast to the solution we outlined to bind both the enciphered PIN block and the *ARQC* in the Segmented Authorisation method, in the Unified Authorisation method, we do not introduce a separate solution. This is due to both the enciphered PIN block and the *ARQC* being received by the CIB simultaneously. However, in our construction for all the three solutions in the Unified Authorisation method, we achieve the same required binding.

This is achieved by, the card generation a hash of the enciphered PIB block and including this in the next generated *ARQC*. At the point of examination of the *ARQC*, the CIB generates a hash of the received enciphered PIN block and compares it with the hash recovered from the *ARQC*. If the two hashes match, it gives assurance to the CIB that the corresponding PIN block is associated with the received *ARQC*. This is explained in more detail, when we have explained each of our solution.

In this Section, we presented our three solutions that enhance the security of OPV process especially when it is deployed in the Unified Authorisation method. We also discussed how we achieve the required binding between OPV and the transaction authorisation in all three of our proposed solutions.

In the next subsequent sections, we analyse our proposed solutions, discuss our implementation experience and provide mechanical formal analysis results.

4.4 Analysis

In this Section, the proposed solutions are evaluated for their security and performance. The security of the proposed solutions is analytically evaluated in relation to the attacker's capability. The performance measurements are taken to show the potential penalties for the existing process if they are adopted. We also subject the protocol for mechanical formal analysis and provide results.

4.4.1 Security Analysis

We first provide a analytical analysis of the proposed solutions mainly based on Unified Authorisation method. The analysis carried out here is short compared to the analysis in chapter 3 to avoid any overlaps. A detailed analysis on common points can be found in Section 3.4.1.

In our analysis, we take the adversary capability and operating environment assumptions in to consideration. According to this, the adversary can compromise an intermediary entity but cannot compromise the smart cards, payment terminals, scheme operator, or CIB. The rationale behind identifying the entities that cannot be compromised by the adversary is because, if an adversary can successfully compromise any of these entities then almost no protection mechanism would be strong enough to protect against attacks on the OPV process. A detailed explanation for this rational can be found in Section 3.4.1.

In our proposed solutions, the OPVrq is encrypted to provide end-to-end security. Before sending the PIN related data to the CIB, it is encrypted either by the smart card or the CTPOS. Furthermore, the OPVrp is also encrypted by the CIB using a session key included in the OPVrq by either the smart card or the terminal. The response message also includes a random number generated by the smart card or payment terminal (depending upon which proposed solution is selected), providing assurance of the freshness of the OPVrp message.

In the card based solution where a symmetric key is used for the encipherment, any attempts of replying the OPVrq by an adversary will be detected as the session key used to encipher this message would have expired.

Furthermore, in the solution that uses a asymmetric key (public key of the CIB), the payment card or the CTPOS generates a symmetric session key which is used by the CIB to encipher the OPVrp message. If an old OPVrp is replayed the terminal/payment card would not be able to decrypt the message due to the session key being wrong. As a result any replay attempts of the OPVrp can be detected.

In Section 4.3.3 we also explained why certain attack scenarios are circumvented because both the enciphered PIN block and the *ARQC* are being received by the CIB simultaneously. The OPV Response Message, attack scenario that we discussed in Section 3.2.3 is only valid for the Segmented Authorisation method but the compromise cannot be carried out in the Unified Authorisation method. Here we explain why it is not possible to carry out the attack in the Unified Authorisation deployment method. We would like to direct the reader to Section 3.2.3 for further details related to the steps involved in the compromise.

The main reason why this particular attack scenario cannot be carried out in the Unified Authorisation method is because the CIB receiving the PIN block and the ARQC in the same message during a EMV transaction. In the Segmented Authorisation method, when the terminal requests the card to generate the ARQC, it also sends details of successful OPV result. Following the terminal's request, when the card generates the ARQC, the fact that OPV has been successfully carried out beforehand is indicated in the ARQC. Furthermore, the card also includes a hash of the enciphered PIN block in association to the particular transaction which acts as a binding between the PIN block and the ARQC.

As opposed to the ARQC generated in the Segmented Authorisation method, in the Unified Authorisation method, the card indicates in the generated ARQC that OPV is not carried out. In addition to this, the card also includes a hash of the enciphered PIN block in the generated ARQC. With these details, the CIB gets to know that OPV is not carried out by examining the ARQC, hence the CIB always carries out OPV before proceeding to transaction authorisation. Due to this reason, the adversary at the compromised intermediary cannot change the OPV response message. The adversary also cannot remove the PIN block part of the request message and only send the ARQC to the CIB as during examination of the ARQC the CIB will indeed get to know that OPV has not been carried out.

As explained above, we provide binding between the OPV and online transaction authorisation in all three proposed solutions by including the authorising entity generated *Authoriser Unpredictable Number* sent back in the PIN Result Block in the online transaction authorisation. This lets the CIB link a particular OPV to its associated online transaction authorisation.

The solutions also prevent "Yes" card [146] attacks by using a symmetric key generated by the payment terminal. More details related to this can be found in Section 3.4.1. Furthermore, we consider another concern that could raise some security concerns. That is the use of a standard Initialisation Vector (IV) [138] for the symmetric cryptosystem. We detail how our solution overcomes this by using recommended block size and random numbers in Section 3.4.1.

In the next section, we discuss our implementation experience and approach taken.

4.4.2 Practical Implementation

In this section, we describe our implementation of the proposed solutions with the aim of providing potential performance penalties the existing OPV process has to bear if our improvements are adopted. For comparison purposes, we provide performance measurements of both Segmented Authorisation method and the Unified Authorisation method in Table 4.7.

[1]Measurement was taken from a desktop computer.
[2]Measurement was taken from a desktop computer.

Table 4.7: Performance Penalties associated with Each Solution

Proposed Solutions	**Basic Encryption**		**Encrypt-then-MAC**		**Authenticated Encryption**	
	Java Card 1	**Java Card 2**	**Java Card 1**	**Java Card 2**	**Java Card 1**	**Java Card 2**
Performance penalties imposed by proposed solutions on the segmented authorisation method						
Symmetric Key Card	64ms	86ms	138ms	156ms	122ms	136ms
Asymmetric Key Card	138ms	159ms	196ms	218ms	-	-
Asymmetric Key Payment Terminal[1]	34ms		48ms		-	
CIB OPVrp	16ms		28ms		22ms	
Performance penalties imposed by proposed solutions on the unified authorisation method						
Symmetric Key Card	71ms	92ms	143ms	160ms	128ms	141ms
Asymmetric Key Card	143ms	164ms	202ms	225ms	-	-
Asymmetric Key Payment Terminal[2]	36ms		52ms		-	

We use the same test-bed described in Section 3.4.2, for our implementation. Table 3.7 lists the performance measurements given in milliseconds (ms).

Our implementation does not emulate the complete EVM specifications but only implements the proposed improvements. The performance measurement should be taken as an additional execution cost that the EVM process has to bear to implement the proposed solutions in this chapter.

In the next Section, we subject our proposed solution to mechanical formal analysis.

4.4.3 Mechanical Formal Analysis

In this section, we subject the proposed modifications to a mechanical formal analysis based on the CasperFDR tool.

The CasperFDR mechanical analysis framework can be used to test the soundness of a security protocol under a set of defined security properties. We do not evaluate the solutions related to the Unified Authorisation method, as they only differ in hash generation and verification. Whereas, the remaining aspects are more or less the same as for the evaluated protocol in the Segmented Authorisation method. The attacker model and a detailed explanation during the construction of the Segmented Authorisation method of the formal analysis can be found in Section 3.4.3. The listed specifications are defined in the #Specification section of appendices A.1.1 and A.1.2 After successfully evaluating the proposed protocol, the CasperFDR tool did not find any feasible attacks and weaknesses related to the protocol.

4.5 Summary

In this section, we conclude our discussion and summarise the key contributions of this chapter.

This Chapter is an extension to our contributions in Chapter 3 which investigated the Segmented Authorisation OPV deployment method and proposed solutions. The second OPV deployment method which we investigate in this chapter is referred to as the Unified Authorisation method. We then described how certain aspects of this deployment method open up a potential route for an adversary to compromise payment transactions for fraudulent financial gains. Following this, assumptions related to the payment network's operating environment, the capabilities of an adversary and potential attack scenarios were outlined.

Subsequently, we proposed three potential ways to enhance the security of OPV when deployed in the Unified Authorisation method. We also discussed how we bind the OPV process with the online transaction authorisation. Proposed solutions were then analysed for their security and results of potential performance penalty incurred by our proposals were listed. Furthermore, the protocol was then subjected to the mechanical formal analysis using the CasperFDR tool.

The attacks we discuss in this Chapter can be considered to be difficult to detect whether an OPV-based transaction was actually made by the cardholder or the adversary, as the compromise of the intermediary nodes might not be detected in time. Therefore, we consider this to be a concern and suggest that a mandating roll-out of an OPV process in a geographical region should take into consideration these concerns and our potential solutions.

Chapter 5

Tokenisation based Protocol for Offline Payment Transactions

Contents

In this chapter, we focus our attention on Transaction Authorisation which is another important aspect of the EMV payment architecture. We first introduce the EMV Primary Account Number (PAN) based payment architecture and outline an inherent weakness associated with the payments. We then introduce EMV Tokenisation which has been adopted to solve this problem and further identify a bottleneck in this new architecture. Afterwards, we propose a solution that extends the usability of tokenised payments in offline environments. Finally, we mechanically analyse our proposed protocol and implement the solution to obtain performance measurements.

5.1 Introduction

Unlike a contactless smart card, a mobile[1] has additional capabilities including increased computing ability, a greater variety of accessible Application Programming Interfaces (API), and readily available communication channels via a Mobile Network Operator (MNO) or Wi-Fi. Furthermore, modern mobile devices typically feature NFC and hardware or software Secure Element (SE) technologies that provide secure execution environments in which to execute sensitive applications [3, 19, 21]. Hardware SEs provide a secure storage environment for credentials and offer tamper resistance against physical attacks [11]. Mobile payment applications provide additional features including having a number of virtual contactless payment cards issued by different financial institutions in one place; passcode unlocking mechanisms to access virtual payment cards; and the ability to block such cards if a mobile is lost or stolen.

The above properties and features of mobile devices are beneficial for running mobile-based payment solutions. Due to the additional capabilities of the mobile-platform, a payment application that runs on a mobile platform can be integrated with more advanced features compared to a payment applet that runs on a smart card.

The common parameters in which a mobile device that runs a contactless payment application and a payment terminal communicates are standardised under the EMV contactless specification [24]. This provides interoperability between participating payment terminal and contactless mobile payment applications.

An introduction to EMV Tokenisation was given in Section 2.1.4. EMV Tokenisation is increasingly being adopted by the payments industry to prevent PAN compromise as explained in Section 2.1.3 of the thesis.

5.1.1 Problem Statement

The EMV Tokenisation Specification details the requirements for supporting payment tokenisation in EMV transactions [20]. Even though tokenisation provides security against PAN compromise, there are many challenges yet to be addressed in the tokenisation landscape [83]. The lack of support for making or accepting tokenised payments in an offline transaction environment is a shortcoming in this payment architecture.

Finding a solution to this shortcoming is the main focus of this chapter. The current tokenisation architecture requires online connectivity on the terminal in order to reach the payment authorisation entity during a tokenised transaction. However, it is not always possible to have online connectivity in certain transaction scenarios. In

[1]In this chapter, the payer's contactless mobile payment device that emulates a contactless smart card is referred to as the mobile.

this chapter, we identify three scenarios where a fully offline transaction capability is considered beneficial for both the merchant and the consumer.

1. Connectivity is not possible due to the geographical location of a transaction, such as purchases made on aeroplanes and underground subway systems.

2. Steady/continuous connectivity is not guaranteed. If a business is operating in a non-stationary environment, it is most unlikely that the merchant's portable payment acceptance terminal has continuous connectivity to the payment network; for example, a merchant who sells snacks on a fast-moving train using a portable payment terminal. If the merchant is able to accept payments from customers wanting to make tokenised payments, accepting and storing tokenised payments has significantly lower financial loss associated with payment card breaches as we explained in Section 2.1.4. Further more, this may also help improve the merchant's turnover. More significantly, the consumers would be protected by the security of being able to make tokenised payments.

3. It is significantly cheaper to carry out offline transactions due to the communication and processing costs involved with establishing each online transaction individually. An example is carrying out a number of transactions offline and then forwarding all the transactions simultaneously for batch payment processing at a later time.

From the above discussion and example scenarios, the inability to make/accept tokenised payments in an offline environment may act as a deterrent for both consumers and merchants. This could hinder the potential adoption of contactless mobile payment solutions based on tokenisation with in the payments industry.

5.1.2 Contributions

In this chapter, we propose a contactless mobile payment protocol based on EMV tokenisation that allows offline token payments when no online connectivity is present on either the terminal or the mobile. The proposed solution also provides end-to-end encryption between the secure element of the mobile and the terminal. This provides security for transaction data other than the token. The protocol is analysed against protocol objectives and subjected to mechanical formal analysis using Scyther. In our analysis, we show that while tokenised payments prevent PAN compromise during transactions, they are still susceptible to token relay attacks. A discussion related to token relay attacks is carried out in Section 5.4.1. We then discuss how the proposed protocol can be extended to detect and prevent potential token relay attacks using ambient sensing. Finally we implement the protocol and provide performance measurements. The main contributions of this chapter are the following:

1. The protocol introduces the *Offline Transaction Token (OTT)*, providing the ability to make fully offline tokenised payments
2. End-to-end encryption between the secure element of the mobile and terminal provides additional security for transaction data other than the Token

The remainder of the chapter is structured as follows. In Section 5.2, the prospective offline operating environment and the adversary's capability are discussed. The proposed protocol is presented in Section 5.3. A security analysis of the protocol is carried out in Section 5.4 and the protocol is subjected to mechanical formal analysis in Section 5.4.2. The practical implementation of the protocol and performance measurements are given in Section 5.4.3. Finally, in Section 5.5, the discussion is concluded and further research directions are identified.

5.2 Offline Operating Environment and Adversary's Capability

The current EMV tokenisation online operating environment was presented in Section 2.1.5 of the thesis. A generic payment architecture and the transaction message flow for a tokenised contactless mobile payment was illustrated in Figure 2.2 in Section 2.1.5. In this section, we briefly discuss the prospective offline operating environment in order to provide offline payments based on EMV tokenisation. Then the capabilities of the adversary in this environment and potential risk concerns are discussed.

In the offline transaction environment considered in this chapter, online connectivity to reach the authorising entity is not available on either the secure element or the terminal. The secure element and the terminal are the only two parties involved during the transaction. Therefore, in such a scenario the terminal needs to decide whether to accept or decline a tokenised transaction. As the payment transaction is carried out offline, it is paramount to secure the payment and the communication between the secure element and the terminal. The payment settlement phase is carried out when the terminal has online connectivity at a later time.

Taking the prospective offline-based operating environment as discussed above into consideration, the capabilities of the potential adversary who may compromise the tokenisation-based payment system are listed below:

- Cannot break the standardised (strong) encryption algorithms.
- Has the capability to eavesdrop unencrypted messages passed between the secure element and the terminal.
- Cannot compromise the terminal's public key certificate introduced in Section 5.3.1. In Section 5.4.2, we discuss why the terminal's public key certificate is not in the adversary knowledge in the adversary model.
- Cannot compromise the secure element, terminal, scheme operator, token service provider or the issuing bank.

Based on the adversary's capability and the current operating environment, an adversary who compromises token transaction data during an offline transaction scenario may carry out fraudulent transactions. The risk scenarios involved may include the adversary changing the transaction amount to a new value, capturing the OTT, or replaying the same OTT to carry out multiple offline transactions with different terminals. Therefore, as well as providing offline token transaction capability, it is vital to secure the sensitive token transaction data communicated between the secure element and the terminal.

In order to store the OTT securely on the mobile phone for offline use and to prevent the OTT being compromised, tamper resistant storage and secure execution are needed. In our threat model for the complete offline payment environment, the OTT needs to be protected from the adversary. A mobile phone with a card emulation environment such as Host Card Emulation (HCE) does not offer tamper resistance [38, 180, 158]. Therefore, we consider a mobile phone with an embedded secure element which offers both security guarantees in our proposal. In the next section, we take these concerns into consideration and propose our protocol.

5.3 Proposed Solution

In this section, an offline contactless mobile payment protocol based on EMV tokenisation is proposed. The payment protocol is used to make offline payments when there is no online connectivity on either the terminal or the mobile during a tokenised transaction. The objectives of the proposed protocol are listed below.

1. The protocol should be able to make secure offline payment transactions.
2. End-to-end encryption should be provided between the SE and terminal. This offers additional security to protect transaction data other than the Token.

5.3.1 Protocol Assumptions

The assumptions made in the proposed solution are listed below:

- The Token Service Provider (TSP) is a trusted entity that securely generates, issues and de-tokenise transaction tokens on behalf of the bank.
- The TSP in the proposed protocol takes part in the same payment scheme and the TSP generated signatures can be verified by the terminal following the certificate hierarchy shown in Figure 5.2.
- The terminal has been issued with a public key certificate signed by the scheme operator.
- The Offline Transaction Token (OTT) and security-sensitive data including the cryptographic keys are securely stored in the secure element. It is not possible for an adversary to steal/compromise the OTT or data residing in the secure element.
- The nonces generated by the terminal are random and unpredictable. An adversary cannot deduce the second nonce by examining the first nonce.

The notation used in the proposed solution is included in Table 5.1. The tokenised contactless mobile payment architecture of the proposed protocol is illustrated in Figure 5.1. The protocol proposed in this chapter comprises a setup phase, a payment phase and a settlement phase. In the *Setup Phase*, the payment app and the OTT are securely provisioned to the secure element of the mobile. The payment phase can be used to make an offline tokenised payment when there is no online connectivity on either the terminal or the mobile.

5.3.2 Setup Phase

During the setup phase, the personalisation of the payment application and provisioning of OTT related data is accomplished. The provision of payment application and security-sensitive data to the secure element is carried out using a secure Over-The-Air (OTA) channel [18]. Following application personalisation, the payment application's sensitive data elements reside in the secure element and its user interface is located in the mobile platform. The components that reside in the secure element consist of all cryptographic keys needed by the mobile, i.e. S_{mobile} and P_{mobile}. The secure element also stores: $Cert_{bank}(TSP)$, $Cert_{bank}(SE)$, Token Application Transaction Counter ($TATC$), OTT and the token service provider's digital signature on the hash of Static Token Data (STD) which has the notation $sS_{TSP}\left[h(STD)\right]$. The OTT is generated by

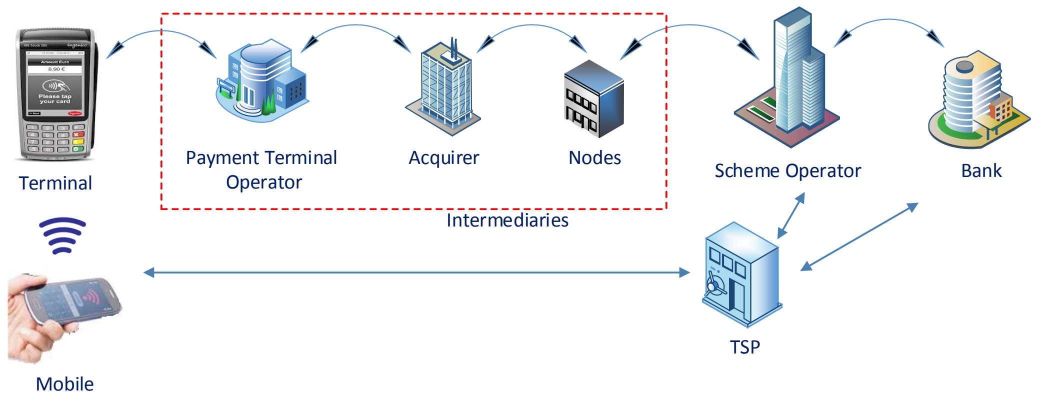


Figure 5.1: Tokenised Contactless Mobile Payment Architecture of the Proposed Protocol

Table 5.1: Notation used in the Proposed Protocol

T/SE/SO/x	:	Terminal/Secure Element/Scheme Operator/Identity of X.
TSP/OTT	:	Token Service Provider/Offline Transaction Token.
TATC	:	Token Application Transaction Counter, count of token transactions since personalisation. It is shared between $mobile, bank$ & TSP and used during key derivations.
$MaxValue$	:	Maximum value of the total offline token transactions allowed per OTT. Predefined value set by the bank.
T_{limit}	:	Transaction Limit is a record kept in the secure element. It is the total value of previous offline token transactions.
$K_{To'}$	:	Token Cryptogram Generation Symmetric Session Key derived by a key derivation function used by TSP.
K	:	SE generated Symmetric Session Key.
$E_K\{Z\}$	:	Symmetric Encryption of data string Z using key K.
S_X	:	Private Signature Key of entity X.
$sS_X\,[Z]$	:	Digital signature outcome (without message recovery) from applying the private signature transformation on data string Z using S_X of X.
P_X, P_X^{-1}	:	Public Encryption/Decryption Key Pair of entity X.
$eP_X\{Z\}$	:	Encryption of data string Z using a public algorithm with P_X.
$Cert_Y(X)$	:	Public Key Certificate of X issued and certified by Y.
a_X	:	Ambient sensor details issued by entity X.
$h(Z)$	:	Hash of data string Z.
n_X / n_{2X}	:	First / second nonce issued by entity X.
$A\|\|B$	:	Concatenation of A and B in that order.

the TSP and has a strong cryptographic binding with the $bank$, TSP, $Token$ and n_{tsp}. The OTT consists of a TokenData part and an encrypted part (Token Cryptogram). The construction of the OTT is shown below.

$$OTT = TokenData||TokenCryptogram$$
$$TokenData = Token||TokenExpiry||TokenR\text{-}ID^{2}$$
$$TokenCryptogram = E_{K_{To'}}\{TokenData||MaxValue\ ||CurrencyCode||n_{tsp}\}$$

The OTT is securely provisioned to the secure element using an OTA channel. Once the OTA is provisioned the secure element keeps a record of the $MaxValue$ and the T_{limit}. The $MaxValue$ is the total offline token transactions value allowed for a particular OTT. This is a predefined value set by the issuing bank, taking the liability involved in offline transactions into consideration. The T_{limit} is the total value of previous offline token transactions carried out using a particular OTT. In a given transaction scenario, the secure element adds the prospective transaction value to the T_{limit} to check whether the combined value exceeds the $MaxValue$. The secure element only presents the OTT to a terminal if the prospective transaction does not exceed the $MaxValue$.

However, whenever there is online capability and the $MaxValue$ of a particular OTT has been reached, a new OTT is provisioned automatically to the secure element using an OTA channel. The provisioning of the new token simultaneously resets the T_{limit}.

Following personalisation of the payment application, the user is required to enter a passcode for secure access to the payment application. The passcode is stored in the secure element for future authentication of the user with the payment application.

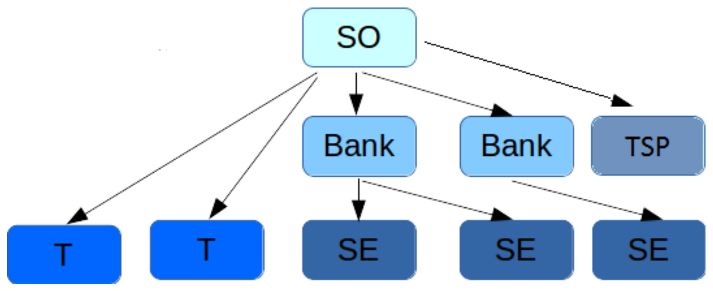


Figure 5.2: Certificate Hierarchy used in the Proposed Architecture

The certificate hierarchy used to verify the public encryption keys and signature verification keys of the entities is shown in Figure 5.2. The scheme operator is at

[2]Token Requester ID: 11-digit unique numeric value, positions 1-3 indicating TSP and positions 4-11 indicating the requester and token domain [20].

the top level of the certificate hierarchy used in the proposal. Terminals and secure elements participating in the payment scheme can verify certificates issued by the scheme operator or entities that have been certified to be trusted in the certificate hierarchy. The TSP also takes part in the payment scheme. As illustrated in Figure 5.2, the secure element's public key is certified by the bank and the terminal's public key is certified directly by the scheme operator. We have constructed the certificate hierarchy in this manner because the number of secure elements that need certifying is greater than the number of terminals. In the current payment architecture, a terminal is deployed by the merchant's acquiring bank or by a subcontractor of the acquiring bank. In our proposal, the scheme operator certifies the terminal's public key. However, on a practical note, in the current payment architecture, terminal manufacturers need to have their terminals certified by a scheme operator. Therefore, it is within the capability of the scheme operator to certify each terminal's public key directly, probably at the same time.

5.3.3 Payment Phase

The protocol messages of the proposal are illustrated in Table 5.2 and explained as follows. To make a contactless mobile payment, the user access the payment application by entering the passcode and holds the mobile device at close proximity to the terminal (the NFC range of the terminal).

Table 5.2: Offline Transaction Token Protocol.

1. $T \rightarrow SE$	:	$t\|\|n_t\|\|Cert_{SO}(T)$
2. $SE \rightarrow T$	:	$eP_T\{se\|\|t\|\|n_{se}\|\|n_t\|\|PDOL\|\|K\}$
3. $T \rightarrow SE$	:	$E_K\{t\|\|se\|\|n_{se}\|\|n_{2t}\|\|amount\|\|CurrencyCode\}$
4. $SE \rightarrow T$	:	$E_K\{se\|\|t\|\|n_{2se}\|\|n_{2t}\|\|OTT\}\|\|sS_{TSP}\left[h(STD)\right]\|\|$ $sS_{SE}\left[h(DAD)\right]\|\|Cert_{bank}(SE)$
		$STD = se\|\|OTT$
		$DAD = n_{2t}\|\|n_{2se}\|\|OTT$
a. T	:	SE read complete
b. T	:	offline token & dynamic data authentication
c. T	:	approved/declined - post-transaction clearing request

Message 1: In the first transaction message, the terminal provides its identity,

terminal-generated first nonce and the terminal's public key certificate to the secure element.

Message 2: The secure element then verifies $Cert_{SO}(T)$ and recovers P_T. Only a genuine terminal can provide a public key certificate that verifies correctly. The second message includes; the identities, n_{SE}, received nonce, the Processing Options Data Object List (PDOL) [24] and the session key. The message is enciphered using P_T before sending to the terminal. The PDOL instructs the terminal what information to transmit back to the secure element.

Message 3: The terminal first deciphers the received message and prepares to send the data requested in the PDOL. The prepared message includes the identities, the secure element's nonce received previously, a fresh nonce, and the amount and the currency code of the transaction. The message is then enciphered using K before sending it to the secure element.

Message 4: The secure element deciphers the previous message and then carries out the following verification steps before constructing message 4:

- The secure element checks whether it has received the expected n_{SE} in order to detect any replay attempts.
- The secure element examines the prospective transaction amount to check it is within the maximum value of a single offline token transaction set by the issuer. If the transaction amount exceeds the maximum value the secure element declines the transaction, otherwise it proceeds to the next verification.
- The secure element then verifies whether the amount of the transaction is with the required limits. This is done by examining the Transaction Limit T_{limit} kept in the secure element's record. The T_{limit} includes the total value of previous offline token transactions. The secure element adds the prospective transaction amount to the value in the T_{limit} and checks to see whether the final amount exceeds the *MaxValue*, which is the total offline token transaction value allowed for a particular OTT. If the *MaxValue* has been reached, the secure element declines the transaction; otherwise the secure element proceeds to construct message 4. The secure element updates the T_{limit} record when message 4 is successfully issued.

The secure element uses the Offline Transaction Token (OTT) for the payment transaction and includes the following data in the constructed message: the identities; n_{2se}; n_{2t}; and the OTT. The message is enciphered using K. The n_{2t} forms part of the Dynamic Application Data (DAD) used by the terminal to detect any replay attempts.

Message 4 also includes the $sS_{TSP}\left[h(STD)\right]$, $sS_{SE}\left[h(DAD)\right]$ and the public key certificate. The $TSP's$ signature on static token data can be used by the terminal to carry out *offline token data authentication* to verify the authenticity of the OTT related data offline. The $SE's$ signature on the DAD can be used by the terminal to carry out *offline dynamic data authentication* to verify that it is communicating with a genuine secure element. Finally the public key certificate can be used to verify the signature through the certificate hierarchy.

Once message 4 is successfully sent, the secure element may leave the NFC field of communication. The terminal then carries out the following four verification steps before the OTT transaction is approved or declined:

- The terminal verifies $sS_{TSP}\left[h(STD)\right]$ by generating $h(SE||OTT)$ and comparing it with the hash recovered from $sS_{TSP}\left[h(STD)\right]$. If the two hashes match, this verifies that the TSP has signed the STD presented to the terminal by the secure element. This provides assurance to the terminal regarding the authenticity of the OTT. If *offline token data authentication* fails, the terminal declines the transaction.

- The terminal then verifies the $sS_{SE}\left[h(DAD)\right]$ produced by the secure element. The terminal generates the hash of the DAD received in message 4 and then compares this with the hash recovered in the $sS_{SE}\left[h(DAD)\right]$. If the two hashes match, *offline dynamic data authentication* is verified successfully, otherwise the transaction is declined due to the potential of a replay attack. A replay of OTT can be detected by the terminal due to a replayed message 4 not having the terminal-generated n_{2t} in the $sS_{SE}\left[h(DAD)\right]$.

If the verification steps are completed successfully, then the terminal approves the offline token transaction. If they fail, then the transaction is declined. In both cases, the outcome is displayed to the user on the terminal and a printed transaction receipt may be produced. The terminal issues a *token payment clearing request* when the terminal is online capable at a later time, following a successful transaction of an offline token payment. In the event of an unsuccessful offline token payment, the terminal declines the transaction, displays a decline message on the terminal and a *token payment clearing request* is not sent. The token payment clearing request starts the settlement phase. The settlement process of the offline token payment may follow the same transaction processing channel as specified in the EMV tokenisation specification [20] and discussed in section 2.1.5. The terminal may forward the token payment clearing requests which include: OTTs and transaction details from a number of transactions, in bulk to the acquirer. The acquiring bank then forwards the request to the scheme operator who then communicates with the TSP. The TSP is able to detokenise and validate the OTTs. The retrieved PAN and transaction details are forwarded to the issuing bank for payment clearing. The acquiring bank is settled via the scheme operator.

5.4 Analysis

In this section, the proposed solution is evaluated to see whether it achieves the protocol objectives. The analysis discusses how the protocol can be extended to prevent token relay attacks.

The analysis takes into consideration the operating environments outlined in Section 2.1.5, protocol assumptions outlined in Section 5.3.1 and those described during the setup stage in Section 5.3.2.

1. **Secure offline payment transactions:** Achieving offline transaction capability during a tokenised payment was the main focus of the chapter. The proposed contactless mobile payment protocol based on EMV tokenisation provides capability of making tokenised payments in a fully offline environment. The proposed protocol, in order to achieve the objective of making offline token payments, uses the OTT which includes STD as payment data.

   The terminal carries out four verification steps to verify whether the offline transaction is genuine. The terminal carries out *offline token data authentication*, by verifying $sS_{TSP}\left[h(STD)\right]$. This gives an assurance regarding the authenticity of the OTT. The terminal does *offline dynamic data authentication* by verifying the $sS_{SE}\left[h(DAD)\right]$. By doing this, a replay of OTT is detected by the terminal as the message would not include n_{2t} if it is not genuine. If these verifications fail, the terminal declines the transaction.

2. **End-to-end encryption to protect transaction data:** The protocol provides end-to-end encryption between the terminal and the secure element. This provides confidentiality by preventing adversaries from eavesdropping on sensitive token transaction-related data during transactions. The end-to-end encryption provides security for transaction-related data other than the token. We further establish this in Section 5.4.2 where we analyse the protocol in Scyther. Scyther did not find any feasible attacks including attacks on the secrecy of transaction data.

5.4.1 Token Relay Attack

Relay attacks during EMV contactless payment transactions have been examined in [102, 103, 122, 95, 169, 70]. Even though tokenisation prevents PAN compromise during an EMV transaction, the current EMV tokenisation architecture specified in [20] does not address concerns about relaying EMV tokenised payments. Consider the example of a token relay attack where a genuine consumer makes a token-based contactless mobile payment at a compromised terminal. The transaction is then relayed to a rogue secure element elsewhere, which makes a payment at a genuine terminal simultaneously.

Investigating relay attacks is not the main focus of this chapter, however, as an additional security feature to prevent or detect potential attempts of relay attacks based on tokenised payments, we carry out the following discussion. We show how our proposed protocol can be extended to detect and prevent token relay attacks by adopting ambient sensing, as discussed and illustrated in Figure 5.3. An introduction to ambient sensing and the explanation of how this can be used is given in Section 5.4.1.

Ambient Sensing

Figure 5.3 illustrates how ambient sensing can be used in the proposed EMV tokenisation-based offline contactless mobile payment protocol to detect and prevent token relay attacks. Previous work related to relay attacks detection using ambient sensing can be found in [136, 175, 100, 178]. Analysis on different sensors and recommendations on selecting different sensors available on mobile devices can be found in [99]. Considering two different ambient environments, AE1 and AE2, a token relay from a genuine mobile in AE1 to a genuine terminal in AE2 via a rogue terminal and mobile can be detected by the genuine terminal or a Trusted Third Party acting as a comparing entity.

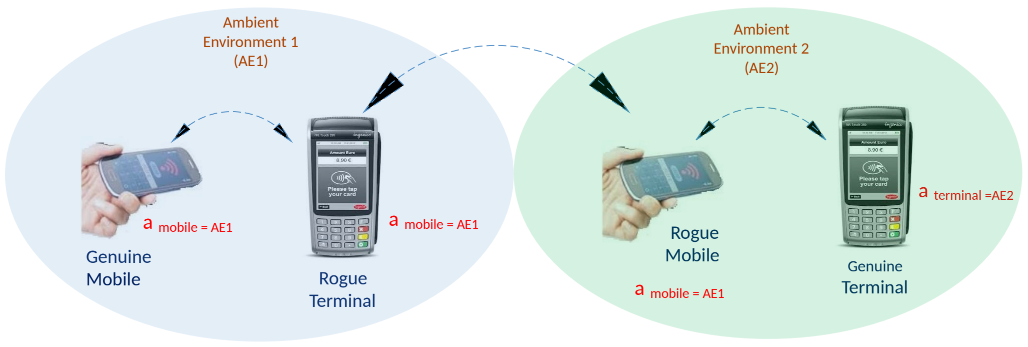


Figure 5.3: Ambient Sensing as a Token Relay Attack Prevention Countermeasure

As illustrated in Figure 5.3, this detection is possible due to ambient sensor data $a_{terminal}$ generated by the genuine terminal in AE2 being significantly different to the relayed a_{mobile} produced by the genuine mobile in AE1. The difference in ambient sensor data is detected when the comparison is made. The attributes collected as ambient sensor data may include atmospheric pressure, ambient noise, ambient light, Global Positioning System (GPS) data, and others [178].

A mobile device and a supported terminal capture their own ambient environment-related data on each device using on-board sensors. In the protocol, the mobile sends its ambient sensor data a_{mobile} in message 4 as shown below.

Table 5.3: Extended Protocol Messages.

4.	$SE \rightarrow T$	:	$E_K\{SE\|\|n_{2se}\|\|n_{2t}\|\|OTT\|\|\mathbf{a_{mobile}}\}\|\|$
			$sS_{TSP}\left[h(STD)\right]\|\|sS_{SE}\left[h(DAD)\right]$
	DAD	$=$	$n_{2t}\|\|n_{2se}\|\|OTT\|\|\mathbf{a_{mobile}}$
b.	T	:	offline token relay detection

In the protocol stage, any attempted token relay attacks are detected offline by the terminal. This is due to the transaction being offline and a trusted third party, such as the TSP, being unavailable to act as a comparing party. To this end, the terminal generates its own ambient sensor data $a_{terminal}$ and compares it with the a_{mobile} received in message 4. This verification can be completed in step b of the protocol. As a_{mobile} forms part of the DAD, it provides data origin authentication of a_{mobile} and other dynamic application data to the terminal by verifying $sS_{SE}\left[h(DAD)\right]$. If the two components match or meet the expected threshold, then the terminal proceeds to the next verification stage; otherwise the terminal declines the offline token transaction because of the potential of a token relay attack.

5.4.2 Mechanical Formal Analysis

In this section, the proposed protocol is subjected to mechanical formal analysis using Scyther [80].

The adversarial model used in this analysis is the Dolev-Yao model in [87]. The following security claims are verified in the analysis: Secrecy of data *(Secret)*, Aliveness *(Alive)*, Weak agreement *(Weakagree)*, Non-injective agreement *(Niagree)* and Non-injective synchronisation *(Nisynch)* [81, 80]. In addition to the claim types defined above, the *verify automatic claims* feature on Scyther was used to verify other claims [80]. In the adversary model, we have excluded the terminal's public key certificate from the adversary's knowledge. This is because only a genuine terminal is able to provide its public key certificate to the secure element and the adversary is unable to compromise a genuine public key certificate that follows a certificate hierarchy.

Following successful execution of the script, the security of data in the claim events were verified and Scyther did not find any feasible attacks. In our analysis, we do not consider the secrecy of the terminal's first generated nonce (nt) sent in Message 1 as a security concern. This is because, according to the protocol assumptions in Section 5.3.1, nonces are random and unpredictable. Therefore, an adversary cannot deduce the second generated nonce ($nt2$) by examining (nt). As a result, knowledge of (nt) is of no value to the adversary because for the construction of the DAD, only the second nonce ($nt2$) is used. The Scyther script is available in Appendix A.2 and can be downloaded from [8].

5.4.3 Practical Implementation

In this section, we provide details of the protocol implementation, our experience, and performance measurements for the protocol.

The protocol was implemented to obtain performance measurements and to provide a comparison with other protocols. In our implementation, a Java application was developed to run as the terminal on a Microsoft Windows 7 PC with a 3.2GHz processor and an 8GB RAM. Then a separate Java card applet was developed to run the payment application on the mobile. The applet was provisioned to the 16-bit hardware secure element of a Nokia 6131 mobile phone. For our implementation, obtaining a mobile phone with an embedded secure element that gave read/write permissions to the secure element was a challenging task. The only mobile phone with an embedded secure element with read/write access to provision our payment applet that was available at the time was the Nokia 6131 mobile phone. In our applet development phase, we found that Java card frameworks v2.2.2 or above were not supported by the secure element. In order to provide compatibility with the secure element, the Java card applet was compiled using Java card framework v2.2.1.

All four messages of our proposed protocol detailed in Table 5.2 were implemented. The communication between the terminal and the secure element was carried out by command and response Application Protocol Data Units (APDU) [160, 191]. In our implementation for asymmetric encryption, we used plain RSA [176] with 1024-bit key and recommended padding. We used MD5 [167] as the hashing algorithm and the RSA Digital Signature Algorithm for signatures [46].

During our implementation, we found that even though the Java card framework v2.2.1 specification [10] supports Advanced Encryption Standard (AES), the secure element we used did not support AES. Because of this limitation on the secure element, we used double-length key Triple DES [121] in Electronic Codebook mode for symmetric encryption. The algorithm was also supported by the Java card framework v2.2.1 [10]. It is important to note that, double-length key Triple DES is an approved cryptographic algorithm and electronic codebook is an approved mode of operation in the EMV specification [14, see: p135-145].

The protocol is scalable to use other advanced algorithms, modes of operations and hashing algorithms. If the implementation had been carried out on a modern mobile phone with an embedded secure element, we would have used AES for symmetric encryption and SHA256 as the hashing algorithm.

Table 5.4: Performance Measurements Comparison in Milliseconds

Measures	**Proposed Protocol OTT**	**SSL [193]**	**TLS [194]**	**P-STCP [34]**	**STCP [36]**
Specification	16bit	32bit	32bit	16bit	16bit
Time to complete	3971ms	4200ms	4300ms	4344ms	3875ms

A performance measurements comparison between the proposed OTT protocol and some other protocols implemented on smart cards is shown in Table 5.4. At the time of writing, we could not find any offline token protocols or their timing measurements in published literature to enable us to carry out a more accurate comparison. In order for us to understand the performance of the protocol, we choose performance measurements from four different protocols available in the literature. The chosen protocols were also implemented on smart cards for measurements in their corresponding papers.

Three different timing measurements were obtained from our implemented protocol. We recorded timing measurements for Message 2, Message 4 and the total time for the protocol to complete. For Messages 2 and 4, timing was measured from the time the command APDU [160] was sent from the terminal to the time it received the response APDU [160] from the secure element. The overall protocol completion time was measured from the time the applet selection command APDU was sent from the terminal to the time it completed all the verifications after receiving Message 4. The measurements for Message 2 and Message 4 were 751 milliseconds and 3086 milliseconds respectively. The overall protocol completed in 3971 milliseconds. The overall time it took to complete the protocol fell in the same performance range as the compared protocols in Table 5.4. Another point to note is that, the 16bit hardware secure element on the Nokia 6131 mobile phone used in our implementation was released in 2006. A more recent 32bit secure element on a modern smart phone would most probably give improved performance measurements.

5.5 Summary

In this chapter, an OTT protocol based on EMV tokenisation was proposed. The proposed solution achieves the two main objectives of the protocol: to be able to make secure offline token transactions and to provide end-to-end security between the terminal and the secure element. The proposal was analysed and we further identified how the protocol could be extended to prevent potential token relay attacks. Finally, we subjected the protocol to mechanical formal analysis using Scyther and provided performance measurements from a practical implementation. At the time of writing, apart from [77, 152], there is no publicly available academic research based on EMV tokenisation. To the author's knowledge, the work carried out here is the first to propose an offline transaction token protocol with mechanical formal analysis, practical implementation and performance measurements.

Chapter 6

Improving Tokenised Payment Security with Dynamic Transaction Tokens

Contents

In this chapter, we further investigate the current tokenisation architecture and identify a number of weaknesses that pose a few security threats to tokenised payments. We identify five potential attack scenarios in the current architecture, especially when a particular implementation of EMV tokenisation uses a static-token which is passed onto payment terminals during every transaction. We then propose a contactless payment protocol that addresses these security concerns and enhances the security of tokenised payment transactions.

6.1 Introduction

EMV provides a number of benefits for both merchants and consumers such as: providing the capability of making a payment transaction at a supporting terminal anywhere in the world. However, in Section 2.1.3, we discussed why, compromising the Primary Account Number (PAN) sent during EMV transactions to be used in card-holder not present or magnetic-stripe transactions is a problem. EMV Tokenisation was adopted as a countermeasure to PAN compromise[20]. Tokenisation replaces the PAN by a substitutive value called the Token which is a 13-19 digit numeric value that does not reveal the PAN and passes validation checks set by the payment scheme[20]. Since its introduction, EMV tokenisation has seen early adoption in contactless mobile payment applications[22, 83, 25].

Near Field Communication (NFC) modules in smart phones and portable devices enable users to carry out close proximity communication which also include contactless payments. Here the mobile emulates a contactless smart card. In this chapter, the payment device that emulates a contactless smart card is referred to as a mobile. Mobiles let users store a number of payment applications in one place and have hardware or software secure element technologies. Secure elements provide a secure execution environment to carry out sensitive executions. Compared to a contactless smart card, one of the additional capabilities of a mobile is the readily available communication channels via the network operator or Wi-Fi.

In Section 2.1.4, an introduction to EMV tokenisation was given and the tokenised payment architecture was discussed. The Scheme Operator (SO) was referred to as the Payment Network in Chapter 2.1.4. The numerous intermediaries involved in the payment communication channel between the terminal and the SO include: third party terminal providers that the acquiring bank may have used as sub-contractors to process merchant payments, nodes that carry out key-translation at different locations in the payment channel. The payment architecture and the transaction message flow of a generic EMV contactless mobile transaction based on tokenisation is illustrated in Figure 6.1. The intermediaries are outlined in a dotted rectangle.

We explained the transaction message flow during a generic EMV tokenised payment transaction in Section 2.1.5. The only additional message flow that we elaborate upon here is the communication between the intermediaries. After receiving the token and token related data, the terminal sends the additional token related data in the transaction authorisation message to the Scheme Operator (SO)/ Card Issuing Bank (CIB) via a number of intermediaries for approval. These intermediaries also engage in same key-translation process.

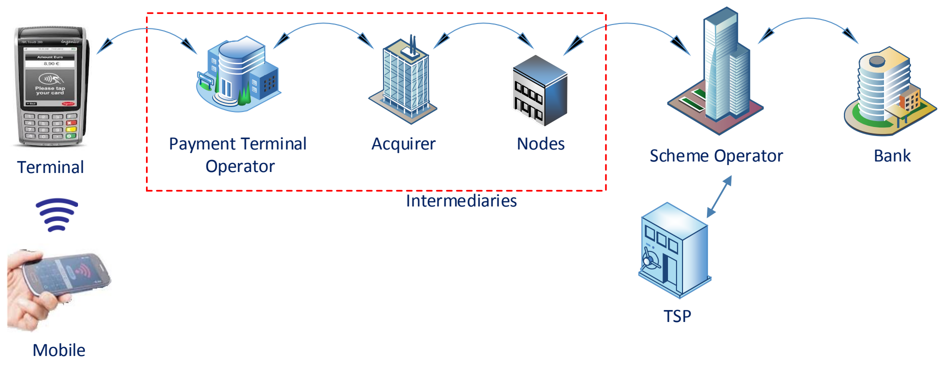


Figure 6.1: Generic EMV Tokenised Payment Architecture

In the payment architecture, the SO has a direct communication channel to the TSP and the bank. The payment terminal operator supplies terminals or rents out terminals to a number of merchants. It also engages in collecting transactions originating from the merchant's terminals and forwards them towards the SO/CIB. A payment terminal operator can either be a third party, or an acquirer's subcontractor. However, whether it's a third party payment service provider or an acquirer's subcontractor does not change our attack scenarios discussed later. A key translation mechanism is used for communication of transaction data between the terminal and the SO in both directions. The same communication path between the terminal and the SO that was taken to send the transaction authorisation request is also taken in reverse to send the transaction authorisation response back to the terminal.

The bank, SO, mobile and TSP are considered as secure and trusted entities. In contrast to this, we consider that the terminal has the potential to be compromised. This is evident from reports and research shown in [183, 74, 89, 60]. The compromised terminal could also represent a rogue NFC enabled mobile phone acting as a terminal [91]. We also consider that the intermediaries have the potential to be compromised. This assumption is not too far fetched due to reports and research shown in [45, 117, 118, 114] which makes our assumptions reasonable. Taking this operating environment into consideration, we expand the discussion to outline potential attack scenarios in the next section.

6.1.1 Problem Statement

In this section, we discuss two problem areas that raise concerns about the security of tokenised contactless mobile payments. The first problem area that we identify is related to the process where the terminal and the mobile is authenticated during a transaction. During a payment transaction, the terminal always authenticates the card/mobile but the card/mobile does not authenticate the terminal. There is a general assumption that the terminal is a trusted and a secure device. An adversary at a rogue terminal can carry out a number of attacks during a tokenised payment transaction because of this lack of mutual authentication.

The second problem area we identify in our work is the fact that similar indelible trust assumptions are placed on the intermediary entities between the terminal and the SO / CIB in the EMV payment architecture. When this trust assumption is disregarded, an adversary compromising one of the intermediary entities is able to compromise payment transaction details and carry out fraudulent transaction. The acronyms used in this chapter are listed in Table 6.1.

Table 6.1: Acronyms used in the Chapter

ARC	:	Authorisation Response Code
CDA	:	Combined Data Authentication
CIB	:	Card Issuing Bank
DDA	:	Dynamic Data Authentication
DTD	:	Dynamic Token Data
DTT	:	Dynamic Transaction Token
EMV	:	Europay MasterCard Visa
NFC	:	Near Field Communication
PAN	:	Primary Account Number
SDA	:	Static Data Authentication
SO	:	Scheme Operator
SPDL	:	Security Protocol Description Language
TAR	:	Token Authorisation Request
TSP	:	Token Service Provider
TVR	:	Terminal Verification Result

6.1.2 Contributions

The chapter provides three main contributions. These are: 1) Providing mutual-authentication between the terminal and the mobile in the proposed solution, so that both entities are able to authenticate to each other. 2) To address the security concerns that arise form using static-token, the proposed solution uses a Dynamic Transaction Token (DTT) that is unique to a particular transaction. 3) To eliminate the indelible trust assumptions placed on the intermediaries, the proposed protocol provides end-to-end encryption between the terminal and the Token Service Provider (TSP) as well as the terminal and the mobile.

The rest of the chapter is structured as follows. In Section 6.2 the two potential problem areas and the corresponding attack scenarios are discussed. The proposed protocol is introduced in Section 6.3 and evaluated in Section 6.4 against protocol objectives. Finally the protocol is subject to mechanical formal analysis in Section 6.4.1.

6.2 Potential Attacks

In this section, we outline potential attack scenarios associated with two main problem areas in tokenised contactless mobile payments.

6.2.1 Adversary Compromises a Terminal

In this problem area, there are three different potential attacks. The attack scenarios are outlined and discussed in Attacks 1, 2 & 3 given below. The terminal is considered to be a trusted device. Even though, a contactless card/mobile is authenticated to the terminal, the cardholder cannot authenticate or verify the terminal to be genuine device, meaning there is no mutual-authentication between the terminal and the card/mobile. When the trust assumption is taken out, a rogue terminal controlled by an adversary could be included in EMV contactless payment process. For these attacks, we assume an adversary with the following capabilities. An adversary:

- can gain full control of the terminal including what is displayed on screen for the payer.
- can change transaction related details such as the amount.
- cannot break standardised encryption algorithms.
- might collude with another adversary that compromises and controls an intermediary between the terminal and the SO/CIB.

Attack 1: *Over Charging*
In this attack scenario, the adversary fraudulently enters a large payment amount (within the contactless limit) for a transaction but displays the correct purchasing product price on the terminal screen for the consumer. Due to the terminal not being authenticated by the mobile, at the time of making the payment, It is not possible for the mobile to detect whether the terminal is genuine or rogue. Also, the transaction amount is not displayed on the mobile. Therefore the user does not have any alternative option other than to believe the amount displayed on the merchants terminal is true. So the consumer, unaware of the fraudulently over charged amount, continues to make a payment. At the time of writing, there are a number of mobile banking applications

that send Short Message Service (SMS) notifications or in-app push notifications for mobile/card base contactless payments [22, 31, 32]. However, these notifications are generated post payment and not before making a payment.

Attack 2: *Capturing Static Token & Related Data*

The second attack scenario we outline is a static token and token related data capturing attack. In this attack scenario, an accomplice controlling the rogue terminal transacts with a genuine mobile making a tokenised contactless mobile payment. The genuine mobile sends the static token and token cryptogram to the rogue terminal. The accomplice captures the static token, its associated cryptogram and other transaction related data. The rogue terminal may display an authentication failed message on the terminal and refuse purchase for the consumer. The captured details are used by the adversary in Attack 4.

Attack 3: *Capturing The Unpredictable Number*

The attack we describe here is a Unpredictable Number capturing attack. The EMV Specification defines the Unpredictable Number as a "Value to provide variability and uniqueness to the generation of a cryptogram [15]". In this chapter we refer to this as the terminal nonce. Even though the attack is not a direct compromise of the terminal, the captured Unpredictable Numbers are generated by the terminal, hence we list this attack under this category. The EMV tokenisation specification does not specify whether offline data authentication needs to be carried out by the terminal [20]. Because tokenised payments operate in an online setting, at first, it is not apparent as to why offline data authentication is actually needed. However, we highlight why failing to carrying out offline data authentication aggravates the identified security concern. The attack steps are described below.

1. An adversary attempts a payment at a genuine terminal to obtain the unpredictable number generated by the genuine terminal.

2. At the absence of offline data authentication, the terminal is unable to verify whether the payment application related data presented by the mobile is genuine.

3. Therefore, the terminal nonce is sent to the mobile as a challenge to be signed by the mobile in order to carry out dynamic data authentication.

4. The nonce forms part of the dynamic application data which is later signed by the mobile to generate the digital signature expected by the terminal.

5. Soon as the nonce is received by the rogue mobile, the adversary captures the nonce and halts any further communication with the terminal.

In some instances, even if Static Data Authentication (SDA) is carried out by the terminal, it may still be possible to compromise the terminal nonce if SDA is not carried out before the nonce is sent. For example, as explained in [70], Visa's payWave qVSDC protocol sends the terminal generated nonce before SDA. This would enable an adversary to obtain the terminal unpredictable number. Potential attacks and other security concerns related to compromising terminal unpredictable numbers are shown in [59, 60].

6.2.2 Adversary Compromises an Intermediary

In this section, we discuss the second problem area that raise security concerns in tokenised contactless mobile payments. In the current EMV architecture, indelible trust assumptions are placed on the intermediaries between the terminal and the SO/CIB. When this trust assumption is disregarded, an adversary compromising one of the intermediaries has a potential attack scenario to infiltrate transaction details and make fraudulent transactions. The adversary at the compromised intermediary observes all transaction data passing through it, which also include transaction authorisation requests, tokens and token related data. For these attacks, we assume the following adversary's capabilities. An adversary:

- can compromise any of the intermediaries.
- can gain access to transaction data at the compromised intermediary.
- cannot break standardised and strong encryption algorithms.
- cannot compromise smart cards, the SO or the CIB.
- might collude with the adversary that compromises a terminal.

Attack 4: *Adversary Replays An Authorisation Response For Cloned Token Data*
The attack scenario is realised when the transacting terminal fails to carry out adequate offline data authentication method such as Dynamic Data Authentication (DDA) or Combined Data Authentication (CDA) [14], but sends the transaction data for online transaction authorisation. We highlight why failing to carrying out adequate offline data authentication methods such as DDA or CDA aggravates the identified security concern. The adversary at the compromised intermediary is able to observe all transaction data passing through it which also includes: transaction authorisation requests intended for the SO/CIB. The attack steps are described below.

1. The adversary works together with the accomplice, who captured the static token and the corresponding token data in the previously discussed Attack 2.

2. The accomplice chooses a terminal that has an established communication path to the SO/CIB via the compromised intermediary and makes a contactless payment with the captured static token data.

3. The terminal carries out SDA on the presented static token data. As the data were captured from a genuine mobile, the SDA verification at the terminal completes successfully. However, without DDA or CDA where a dynamic signature is generated by the mobile and verified by the terminal, the terminal is not able to detect the cloned data.

4. The terminal sends the transaction data online for transaction authorisation.

5. The adversary, instead of passing the transaction authorisation request to the authorising entity, stops the request from reaching the authorising entity. The particular transaction can be identified by the adversary using the static token included in the message.

6. Instead, the adversary replays an Authorisation Response Code (ARC) pretending to have come from the authorising entity and indicates that the transaction was successfully authorised. Once the authorisation response is received, the terminal approves the transaction.

7. Unlike in a contact-based EMV transaction, the transaction authorisation response cryptogram is not sent to the contactless card/mobile [14]. One of the reasons for this is that in contactless EMV, there is no assurance that the card is kept in the reader's field by a cardholder. Because of this reason, the transaction authorisation response is not enciphered by the bank with a key shared between the card/mobile and the bank.

Attack 5: *Replaying An Authorisation Response For DDA/CDA*

In Attack 4, we explained that the EMV tokenisation specification does not specify whether offline data authentication needs to be carried out [20] as tokenisation operates on an online environment and showed how this was realised when the terminal did not carry out DDA/ CDA as offline data authentication. However, from our understanding, it is still possible to provide offline data authentication for tokenised payments for additional security before a transaction is sent online for authorisation. In this attack scenario, we assume that the terminal is carrying out DDA/CDA and identify another weakness that could lead to a potential compromise. The attack steps are described below.

1. The adversary works together with an accomplice, who is in possession of a number of lost & stolen contactless mobiles. The attack is carried out during the time-slot between the cards/mobiles are lost/stolen and the relevant issuing banks are notified by the owners.
2. The accomplice chooses a terminal that has an established communication path to the SO/CIB via the compromised intermediary and makes a contactless payment.
3. The terminal carries out the dynamic offline data verification. As the dynamic signature is generated by a genuine mobile, the terminal verification finishes successfully. The terminal then sends the transaction data online for authorisation.
4. The adversary, instead of passing the transaction authorisation request to the authorising entity, captures it.
5. The adversary replays a previously communicated ARC generated by the authorising entity. Once the authorisation response is received, the terminal approves the transaction.

6.3 Proposed Solution

In this section, we propose a solution that addresses the security concerns discussed in Section 6.2. The main objectives of the protocol are listed below.

1. Should prevent Attacks 1, 2, 3, 4 & 5.
2. There should be a process to carry out mutual authentication between the terminal and the mobile.
3. End-to-end encryption should be provided between the secure element and the terminal, as well as between the terminal and the TSP.

6.3.1 Protocol Assumptions

We have made the following assumptions in our proposed solution:

- A secure channel is used for the communication between the mobile and the TSP (SO in this instance).

- The TSP is a trusted entity that provides transaction token issuing, de-tokenisation, token updates and management on behalf of the CIB in a secure manner.
- The SO acts as the TSP in the payment architecture discussed in the proposed solution.

The notation used in the proposed solution is given in Table 6.2. The tokenised contactless mobile payment architecture of the proposed protocol is illustrated in Figure 6.2. The proposed protocol has a setup stage and a payment stage. In the *Setup Stage*, involves securely provisioning the payment app and related data to the mobile. The *Payment Stage* is initiated when making a contactless mobile payment. The transaction scenario in this chapter is when both the terminal and the mobile are online capable to reach the TSP. Providing offline tokenised payments is not the focus of this chapter and related work on this was carried out in the previous chapter 5.

Table 6.2: Notation used in the Proposed Protocols

T/SE/x	:	Terminal/Secure Element/Identity of X.
TATC	:	Token Application Transaction Counter, count of token transactions since personalisation. It is shared between $mobile, bank$ & TSP and used during key derivations.
K	:	SE generated Symmetric Session Key.
K_{s1}	:	Symmetric Encryption Session Key shared between TSP and SE.
K_{s2}	:	TSP generated Symmetric Encryption Session Key used by the terminal to communicate with the TSP.
$K_{To'}$	:	Token Cryptogram Generation Symmetric Session Key derived by a key derivation function used by TSP.
$E_K\{Z\}$	:	Symmetric Encryption of data string Z using key K.
S_X	:	Private Signature Key of entity X.
$sS_X\,[Z]$	:	Digital signature outcome (without message recovery) from applying the private signature transformation on data string Z using S_X of X.
P_X, P_X^{-1}	:	Public Encryption/Decryption Key Pair of entity X.
$eP_X\{Z\}$	:	Encryption of data string Z using a public algorithm with P_X.
$Cert_Y(X)$	:	Public Key Certificate of X issued and certified by Y.
$h(Z)$	:	Hash of data string Z.
n_X / n_{2X}	:	First / second nonce issued by entity X.
$A\|\|B$	:	Concatenation of A and B in that order.

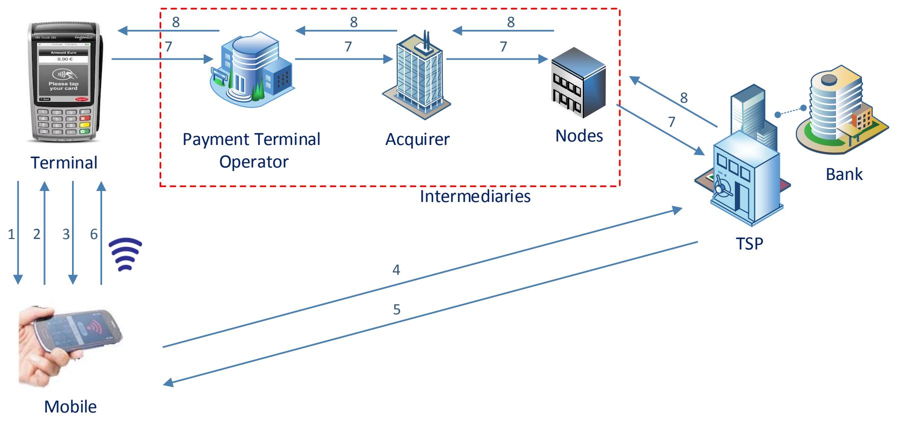


Figure 6.2: Tokenised Contactless Mobile Payment Architecture of the Proposed Protocol

6.3.2 Setup Stage

During the setup phase of the protocol, the personalisation of the payment application and provisioning of security sensitive data elements of the payment application and credentials are carried out using a secure channel. Following the application personalisation, the security sensitive data elements of the payment application reside in the SE and the user interface part of the payment application reside in the mobile platform. The data elements stored in the SE includes, all cryptographic keys needed by the mobile eg: K_{SE}, K_{s1}, S_{SE} & P_{SE}/P_{SE}^{-1}. The SE also stores: $Cert_{bank}(TSP)$, $Cert_{bank}(SE)$, Token Application Transaction Counter ($TATC$). Following successful personalisation of the payment app, on the first use, the user is required to enter a strong pass-code on first access which is then used for future authentication to login to the payment app. The subsequent transaction protocols are constructed based upon the above mentioned data elements.

Terminals and secure elements participating in the payment scheme can verify certificates issued by the SO or entities that have been certified to be trusted in the certificate hierarchy. The TSP also takes part in the payment scheme.

6.3.3 Payment Phase

To make a payment during the payment phase of the protocol, the user opens the payment application by entering the pass-code and taps the device on the terminal. The protocol messages of the proposed solution are illustrated in Table 6.3 and explained as follows.

Table 6.3: Dynamic Transaction Token Protocol Messages.

1.	$T \rightarrow SE$	:	$t\|\|n_t\|\|Cert_{SO}(T)$
2.	$SE \rightarrow T$	:	$V\|\|sS_{SE}\left[h(V)\right]\|\|Cert_{Bank}(SE)$ $V = eP_T\{se\|\|t\|\|n_{se}\|\|n_t\|\|PDOL\|\|K\|\|tsp\}$
3.	$T \rightarrow SE$	:	$W\|\|sS_T\left[h(W)\right]$ $W = E_K\{t\|\|se\|\|n_{se}\|\|n_{2t}\|\|amount\|\|eP_{TSP}\{t\|\|amount\|\|n_{3t}\}\}$
4.	$SE \rightarrow TSP$	:	$E_{K_{s1}}\{se\|\|tsp\|\|n_{2se}\|\|TokenR\text{-}ID\|\|amount\|\|$ $TATC\|\|Cert_{SO}(T)\|\|eP_{TSP}\{t\|\|amount\|\|n_{3t}\}\}$
5.	$TSP \rightarrow SE$	:	$E_{K_{s1}}\{tsp\|\|se\|\|n_{2se}\|\|n_{tsp}\|\|eP_T\{DTD\}\|\|sS_{TSP}\left[h(DTD)\right]\}$ $DTD = tsp\|\|se\|\|t\|\|n_{2tsp}\|\|n_{3t}\|\|DTT\|\|K_{s2}$
6.	$SE \rightarrow T$	:	$E_K\{se\|\|t\|\|n_{3se}\|\|n_{2t}\|\|eP_T\{DTD\}\|\|sS_{TSP}\left[h(DTD)\right]\}$
7.	$T \rightarrow TSP$	:	$Token\|\|E_{K_{s2}}\{t\|\|tsp\|\|Token\|\|n_{2tsp}\|\|n_{4t}\|\|DTT\|\|$ $POSem\|\|TVR\}$
8.	$TSP \rightarrow T$	:	$Token\|\|E_{K_{s2}}\{tsp\|\|t\|\|n_{3tsp}\|\|n_{4t}\|\|Token\|\|$ $TokenAssuranceLevel\|\|\ PANlast4digits\|\|ARC\}$

Message 1: At the start of the protocol, the T provides its identity, n_t and $Cert_{SO}(T)$ to the SE.
Message 2: The SE obtains P_T after verifying $Cert_{SO}(T)$. The SE constructs a message that includes: both identities, n_{se}, n_t, the Processing Options Data Object List (PDOL) that instructs the T what information to send back to the T[24], a session key generated by the SE to be used in further communication between the T and the identity of the TSP. The SE enciphers the message using P_T. A digital signature of the message is generated by the SE and both the enciphered part and the digital signature is sent to the T. The SE's public key certificate is also sent in the same message.
Message 3: Using the certificate hierarchy, the T verifies the signature of the secure element in order to authenticate the device that is making the payment. Once the SE is authenticated, the T deciphers the message. Afterwards, the T encrypts and signs a message which includes: the identities, n_{se}, n_{2t}, the amount of the transaction and an encipherment carried out using P_{TSP} on the T's identity, amount and n_{3t}. The full message and the signature is then sent to the SE.
Message 4: Using the certificate hierarchy, the SE verifies the signature, authenticates

the T and deciphers message 3 to obtain transaction related information. Now the SE prepares to request a token from the TSP by constructing a message. The message includes: the identities, n_{2se}, Token Requester ID, amount, the Token Application Transaction Counter (TATC), $Cert_{SO}(T)$ and $eP_{TSP}\{t||amount||n_{3t}\}$. The message is then encrypted using the symmetric key K_{s1} shared between the TSP and the SE before sending.

Message 5: The TSP, first verifies the $Cert_{SO}(T)$ and obtains the T's public key. The TSP deciphers $eP_{TSP}\{t||amount||n_{3t}\}$ and checks whether the amount recovered from this matches the amount requested by the SE. If satisfied, the TSP queries the CIB and verifies the user eligibility to be issued a new token. Once this process is carried out, a DTT and a session key K_{s2} is generated by the TSP. The TSP then creates Dynamic Token Data (DTD) which includes: the identities, n_{2tsp}, DTT and K_{s2}. The TSP signs the hash of DTD. The DTD is then enciphered using P_T. The TSP then creates a message that includes: the identities, n_{2se}, n_{tsp}, $eP_T\{DTD\}$ and $sS_{TSP}\,[h(DTD)]$. The message is then enciphered using K_{s1} before sending. DTT is constructed as follows;

$$\begin{gathered} DTT = TokenData||TokenCryptogram \\ TokenData = TokenID||TokenExpiry||TokenR\text{-}ID \\ TokenCryptogram = E_{K_{To'}}\{TokenData||amount||n_{3t}\} \end{gathered}$$

Message 6: The SE, prepares a message to send the $eP_T\{DTD\}$ and $sS_{TSP}\,[h(DTD)]$ to the T. The message includes: the identities, n_{3se}, n_{2t}, $eP_T\{DTD\}$ and $sS_{TSP}\,[h(DTD)]$. The message is then enciphered using K before sending. If the SE is not in the NFC field, the user taps the SE on the T again to transmit the message. Once the message is successfully sent to the T, the SE may leave the NFC field.

Message 7: After deciphering the message received from the SE, first the nonce is examined by the T to detect any replay attempts. The T then deciphers the $eP_T\{DTD\}$ to obtain DTD and verifies $sS_{TSP}\,[h(DTD)]$ to have been generated by the TSP. Once satisfied, the T carries out *dynamic token data authentication* to verify the authenticity of the presented data. For this the T generates the hash of the DTD received in the previous message and compares this with the hash recovered in the $sS_{TSP}\,[h(DTD)]$. If the two hashes match, *dynamic token data authentication* is verified successfully, otherwise the transaction is declined due to the potential of a replay attack.

Depending on the outcome of the *dynamic token data authentication*, the T constructs a Token Authorisation Request (TAR) and forwards it to the TSP for payment authorisation. To construct the payment authorisation message, the T first constructs a message which includes: identities, Token, n_{2tsp}, n_{4t}, DTT, Point-Of-Sale Entry Mode (POSem)[1] and the Terminal Verification Result (TVR) indicating the outcome of the offline dynamic token data verification. This message is then enciphered using K_{s2}. The T also appends the $Token$ to the encipherment before forwarding the message to the TSP. The existing key translation mechanism is used by the T to forward the TAR to the TSP via the Intermediaries, for financial transaction authorisation.

The only data sent in the clear is the $Token$, which on its own cannot be used by the $Intermediary$ to obtain any useful information corresponding to the PAN. In the operating environment of the proposed solution, the SO acts as the TSP, therefore the message is received at the TSP. However in a scenario where the scheme operator is not taking the role of the TSP, the scheme operator by observing the Token can identify which TSP it needs to forward the token to.

Message 8: After receiving the TAR, the TSP carries out the following checks to validate the token:

- queries its database records in-relation to the issued tokens and checks details such as: expiry, requester ID, amount and the token cryptogram.
- if the token related data is validated properly, the TSP conducts payment token de-tokenisation to map the token details into PAN details.

Following these verifications the TSP retrieves the PAN details and contacts the CIB to obtain an ARC. The TSP provides information such as: the PAN, PAN expiry date, amount, POSem, token, token expiry, token requester ID and the Token Authorisation Request Result (TAR_{result}) in order to obtain the ARC. The TAR_{result} contains three main components. These are: the outcome of TSP's token verification has passed or failed, $TokenAssuranceLevel$ which indicates the level of assurance that the TSP has assigned to the token depending on the confidence of the TSP and $TokenAssuranceData$ which indicates the data used by the TSP to assign a token assurance level. The CIB before issuing the ARC carries out the following account level validations:

- retrieve account details corresponding to the PAN.
- check whether there are sufficient funds available and no account restrictions.
- verify POSem and the token has not been presented for authorisation before.

[1]The POSem acts as a Token Domain Restriction Control [20] to prevent other cross channel fraud by restricting the tokens to a specific payment channel (contactless mobile payments in this scenario).

- check the outcome of the TAR_{result} validation carried out by the TSP.

Following all the validation steps, the ARC is issued to the TSP by the CIB. Afterwards, the TSP constructs a message that includes: the identities, n_{3tsp}, n_{4t} the Token, Token Assurance Level, the last 4 digits of the PAN and the ARC generated by the CIB. The message is then enciphered using K_{s2}. The TSP also appends the $Token$ to the encipherment before the message is sent to the T via the Intermediaries. The Intermediary cannot deduce any information corresponding to the PAN or the authorisation response other then the $Token$.

Once the message is received, the T deciphers the message using the session key and examines the results in order to approve/decline the transaction. The outcome is displayed on the T. The merchant may produce a receipt that includes transaction details such as the amount, last 4 digits of the PAN, date, time and ARC to be given to the user upon request.

6.4 Analysis

In this section, the protocol proposed in this chapter is analysed for its security and protocol objectives. In our analysis we have taken the following into consideration: the operating environments outlined in Section 2.1.5, adversary capabilities outlined in Sections 6.2.1 & 6.2.2 and protocol assumptions outlined in Section 6.3.1.

At the beginning of the protocol, both the secure element and the terminal are authenticated to each other. The established mutual-authentication between the two entities provides a strong security assurance before security sensitive transaction data are communicated. Due to the unforgeability of the digital signature used, only a genuine secure element and the terminal is able to generate their own signatures. The signatures can be verified using the certificate hierarchy.

The proposed protocol provides end-to-end encryption between the terminal and the secure element. This provides confidentiality to token transaction related data by preventing adversaries from eavesdropping. The protocol also provides end-to-end encryption for the communication between the terminal and the TSP which provides confidentiality to the communicated information and eliminates the need for placing indelible trust assumptions on the intermediaries. Below we analyse how the identified attacks that compromise token transaction data in Section 6.2 are prevented in the proposed protocol. Table 6.4 categorises different countermeasures used for each attack.
Attack 1 *(Over Charging):* In the proposed protocol, message 3 sent by the terminal to the secure element has transaction related data including the amount which is displayed on the users mobile. The request to obtain the dynamic transaction token

is requested by the mobile device from the TSP, only after the user authorises the amount displayed on the user's mobile screen. Any attempts taken by the merchant to overcharge the user will be detected and the transaction cancelled. Furthermore, the corresponding token is requested by the mobile using the data received in message 3, hence a rogue merchant is not in a position to change the amount to a different value. Further more, any amendments to the transaction amount can be detected due to the DTD having the transaction amount in it.

Attack 2 ***(Capturing Static Token & Related Data):*** In the proposed protocol, a mutual-authentication process is carried out between the terminal and the mobile device before the token and token related data is given to the terminal. The n_{se} is sent by the mobile device as a challenge in message 2 for the terminal to sign with other related data. After receiving message 3, the mobile verifies the digital signature and authenticates the terminal to be a genuine device. The mobile aborts the protocol if the terminal is not successfully authenticated at this stage.

Furthermore, due to the dynamic nature of the transaction token issued by the TSP, it is unique to a particular transaction and can only be used once. A replay of DTT can be detected by a genuine terminal due to a replayed message 6 not having the terminal-generated n_{3t} in the $sS_{TSP}\,[h(DTD)]$. Any attempts to carry out Attack 2 is prevented by these countermeasures. If a particular transaction token is compromised by an adversary and tried to replay it in another fraudulent transaction, the TSP would not authorise the transaction for the second time. As the mobile requests a DTT for every transaction, the TSP is aware of a transaction even before a payment authorisation request is made by a terminal. This introduces an additional layer of security to prevent unauthorised transactions, as well as facilitating accurate approvals & risk assurance levels for the tokenised payment transaction.

Attack 3 ***(Capturing The Unpredictable Number):*** The terminal can verify the authenticity of the digital signature generated by the mobile device due to the unforgeability of the digital signature algorithm used. In addition to this, any attempts of replaying the digital signature is prevented by the n_t included in the digital signature. This provides an assurance regarding the freshness of the message and the signature. If the verification fails, then the terminal declines the overall transaction which prevents the terminal from generating the third nonce n_{3t} which is used in the DTT. Furthermore, the protocol prevents any malicious entity from compromising the n_{3t} by using end-to-end encryption between the terminal and the SO.

Attack 4 ***(Adversary Replays An Authorisation Response For Cloned Token Data):*** The proposed protocol uses the following countermeasures to defend against this attack. Firstly, at the start of the protocol, mutual authentication is established between the terminal and the secure element. By taking this approach, only if the secure element is authenticated in message 2, the terminal proceeds to the transaction by sending transaction related data in message 3. Secondly, the proposed solution uses a DTT rather than a static token. This means that because the DTT includes both the terminal's and the TSP's nonces which makes the token specifically unique to a particular transaction. A replay of DTT can be detected by the terminal due to a replayed message 6 not having the terminal-generated n_{3t} in the $sS_{TSP}\,[h(DTD)]$. Furthermore, the protocol provides end-to-end encryption for the communication between the terminal and the TSP. This prevents the adversary at the compromised intermediary from replaying an authorisation response back to the terminal and any such attempts are detected by the terminal.

Attack 5 *(Replaying An Authorisation Response For DDA/CDA):* Unlike Attack 4 where only SDA is carried out, this attack scenario is even possible when DDA/CDA is carried out by the terminal. The attack is prevented in the proposed solution by providing end-to-end encryption for the communication between the two entities. This provides confidentiality which prevents the adversary at the compromised intermediary from learning any useful financial payment information related to the communicated data. Furthermore, nonces generated from both the terminal and the TSP are included in messages communicated between each other as well as in the DTT. This makes any replay attempts detectable for the terminal in the event of any authorisation response replay. Furthermore, the payment application needs the user to enter a passcode before use. It must be also noted that, if the mobile device is lost, stolen or damaged, the user is able to inform the CIB in order to restrict access to the mobile app and to deny access to the token requests.

Table 6.4: Attacks and Countermeasures used in the Proposed Protocol

Attack	**Mutual Authentication**	**End-To-End Encryption**	**DTT**	**Other**
1:Over Charging	✓		✓	Amount displayed on mobile
2:Capturing Static Token & Related Data	✓		✓	
3:Capturing The Unpredictable Number	✓	✓	✓	
4:Adversary Replays An Authorisation Response For Cloned Token Data	✓	✓	✓	
5:Replaying An Authorisation Response For DDA/CDA	✓	✓	✓	Passcode for payment app

6.4.1 Mechanical Formal Analysis

In the previous section we analysed our proposed solution against the security and objectives of the protocol. Having carried out this informal analysis, in this section, we subject the protocol to mechanical formal analysis using Scyther [80]. The proposed protocol was modelled and provided as input to Scyther using the Security Protocol Description Language (spdl) defined in[81]. The spdl provides three main protocol modelling features: *roles, events and claims*. The roles define the entities in a protocol, which characterise events. The send and receive operations are classed as *send* and *recv* events respectively; each corresponding *send* and *recv* event has the same sequence number. The security goals and objectives of a protocol that require verification are specified using *claim* events. We used the Dolev-Yao model as the adversarial model used in this analysis[87]. The following security claims are verified in the analysis: Aliveness *(Alive)*, Weak agreement *(Weakagree)*, Non-injective agreement *(Niagree)* Non-injective synchronisation *(Nisynch)* and Secrecy of data *(Secret)* for: DTT, ARC, K, K_{s2} [81, 80].

The script was run on an Intel CORE-i7 2GHz machine with 8GB of RAM. When the security claim events were run together during protocol analysis, Scyther tool was crashing. We identified that the reason for this was the RAM getting full after a few hours of protocol analysis. To overcome this issue, the security claims were analysed one after the other. A point to note that is, running these security claims all together or separately, does not effect the outcome of the security analysis. Following successful execution of the script, the security of data in the claim events were verified and Scyther did not find any feasible attacks within the bounded state space. The Scyther script is available in Appendix A.3 and can be downloaded from[5].

6.5 Summary

In this chapter of the thesis, we extended our discussion of the current EMV tokenised payment architecture. We then identified five potential attack scenarios in two problem areas of the current architecture that would lead to a transaction compromise. A contactless mobile payment protocol based on dynamic tokens was proposed to address the identified security concerns and to guarantee the protocol objectives. The protocol was then analysed for its security and objectives. Finally the protocol was subject to mechanical formal analysis which did not find any feasible attacks within bounds.

Part III

Improvements for Bitcoin/Blockchain based Distributed Payments

Chapter 7

Establishing True Fair-Exchange in Anonymous Bitcoin Payments

Contents

The e-commerce market is growing rapidly as well as the number of transactions carried out each second globally. Another aspect is that, consumers' growing interest in alternative payment methods other than credit/debit card based payments. This is valid in both traditional Point of Sale environments and e-commerce environments. In this chapter, we identify that establishing fair-exchange while using an anonymous payment method such as Bitcoin is difficult. Addressing this issue, we then propose a protocol that guarantees strong/true fair-exchange while preserving the anonymity of the transacting parties when Bitcoin is used as the payment method in e-commerce transactions. Finally the solution is analysed for its security requirements.

7.1 Introduction

Anonymity prevents merchants, payment processors, third party payment providers, subcontractors, adversaries that compromise payment transactions, etc. from learning consumer personal information, spending habits and financial details. This would also help reduce fraudulent activity related to identity theft by not revealing any personally identifiable information during a payment transaction. Currently the majority of payment methods used for e-commerce transactions do not provide guaranteed anonymity for both the customer and the merchant at the same time. Although services such as PayPal can be used to hide personal financial details from the merchants and Tokenisation as we discussed in Section 2.1.4 can be used to replace the PAN with a surrogate value.

Modern technological advancements have given a dramatic boost towards the evolution of the Internet. A strong outcome of this global expansion of inter-networked technology is the emergence of e-commerce. The e-commerce market is growing and is expected to reach $4.058 trillion in 2020 [33]. In the last few decades there have been a significant increase in the number of consumers opting in for online shopping as well as the number of e-commerce transactions carried out globally every day [40, 200].

While Credit/Debit card payment schemes dominate a majority of the e-commerce payment transactions, there are other alternative payment schemes such as Real Time Bank Transfers, Offline Credit, Direct Debit, eWallets, Mobile Payments and Digital Cash which also provide a way to complete payment transactions at present [84, 200].

In a traditional Point-of-Sale (POS) transaction, the parties involved do not have to be concerned much regarding the guaranteed delivery of the purchased product or the guaranteed payment for the sold product and vice-versa. This is due to the fact that the transaction is carried out in a face to face environment in a physical shop. Even though, there is the possibility of misleading or misrepresentation of the physical goods by the seller, adequate checks and visual observations can be carried out to identify such issues. Furthermore, the consumer can make a simple payment by using cash for the goods and services he/she purchased without handing in their personal or financial details to the merchant and other third parties.

However, a buying and selling transaction in e-commerce is much different from the previously explained. In an e-commerce environment there are issues related to anonymity and fairness of transactions. In an e-commerce environment where transacting parties do not see each other physically makes it possible for a dishonest party to misbehave. In one hand, a merchant could simply not deliver the goods to a consumer once the payment is received or claim the payment was never received. On the other hand, a consumer could simply disappear without paying a merchant once goods have been received or deceivingly claim that the content was never received.

Therefore, a merchant may ask a consumer to provide personal identifiable information such as: name, address, contact number, prior payment or even identity copies before a product is delivered. This information would help the merchant for any dispute resolution, take legal action for non payment or to minimise any financial loss. In such an e-commerce transaction, a consumer might lose privacy by having needed to provide personal and financial information to merchants, payment processors, third party payment providers, etc. Furthermore, most of widely used e-commerce payment methods are also linked to personal identity of the account holder. Similarly, a consumer might ask the merchant to provide contact details, business registration documents, etc. that can be used in a dispute resolution for non delivery of product. The consumer may involve his/her financial organisation to establish some sort of liability and accountability of the merchant to prove that a payment was made to the merchant's bank account.

Most of the current electronic payment methods, neither provide anonymity of consumer to protect consumer privacy nor security of financial information to guarantee the security of the transferred value at the same time. Instead, there is a trade-off between these two aspects [84]. The concerns discussed above have led attempts to finding a solution to give e-commerce users the freedom of making anonymous payments without having to reveal personal details and to guarantee fair-exchange [208, 209, 207, 161].

Protocols that help realise anonymity and user privacy during payment transactions are called *Anonymous Payment Protocols* which was introduced and discussed in Section 2.3.1. Protocols that helps to establish fairness in e-commerce transactions are called *Fair-exchange Protocols*. A discussion on fair-exchange protocols and introduction of Weak and Strong fairness in e-commerce were given in Section 2.3.2. A combined solution that would realise fairness as well as anonymity is called an *Anonymous Fair-exchange Payment Protocol*.

In the next section, we extend our discussion on anonymous fair-exchange and identify an issue with using Bitcoin in e-commerce transactions.

7.2 Problem Statement

When we consider industry wide adoption of Anonymous Fair-exchange Protocols, even though protocol design may guarantee anonymity and fairness, when it comes to making anonymous payments in the real world this is a big challenge. This is because of a number of factors such as: a majority of existing financial service providers use

EMV based centralised payment solutions that do not provide anonymity, distributed payment solutions that provide anonymity (cryptocurrencies) are either not recognised by consumers or don't have a market trading value to be accepted as a method of payment by merchants, lack of public awareness and confidence, etc.

In recent years, Bitcoin is one of the few cryptocurrencies that has gone through nearly a decade of maturity and gained increased public interest as an alternative payment method. An introduction to Bitcoin was given in Section 2.2.2 of the thesis. Bitcoin payments are anonymous in the sense that they cannot be linked to a real identity of a user but payments are irreversible. At the time of writing Bitcoin payments cannot be retrieved or cancelled. Due to this reason and payments being anonymous it is difficult to guarantee fairness between two parties engaged in an e-commerce transaction using Bitcoin. This is a major concern for a genuine consumer or a merchant who has or would like to make and receive payments using Bitcoin but is reluctant to do so due to the uncertainty of receiving content for payment or payment for delivered content and vice-versa [144, 61, 107, 204, 112].

Current solutions such as "e-Bay Guarantee" and "PayPal Buyer Protection" for non-Bitcoin transactions aims to provide a level of peace of mind for consumers. However, these methods have capped transaction values or involve lengthy dispute resolutions. For Bitcoin users, however, there are no such options available to give peace of mind in e-commerce payments. Bitcoin has now become a competitive player for alternative payment methods with a considerable market capitalisation of $62 billion as of September 2017 [57]. At the time of writing, there are no specifically designed e-commerce protocols for Bitcoin that addresses aforementioned concerns and the work we carry out here is one of the first Bitcoin-specific proposals that guarantees strong-fairness and anonymity. Our main contribution of this chapter is the proposed fair-exchange protocol that guarantees strong fairness while preserving the anonymity of the transacting parties in section 7.3.

The rest of the chapter is structured as follows. In Section 7.3, we propose our protocol and in section 7.4, a security analysis of the protocol is carried out. In Section 7.5, we add an extension to our protocol to support the improved Zerocash system for Zerocoin. Finally in Section 7.6, we conclude our discussion.

In the next section we present our proposed solution.

7.3 Proposed Protocol

The main contribution of the paper is an anonymous fair-exchange payment protocol. The proposed protocol guarantees fair-exchange and anonymity by using Bitcoin. The proposed protocol realises strong fair-exchange and anonymity within Bitcoin users. The solution keeps the involvement of a TTP to a minimum by using an Off-line TTP. This makes our proposed solution an optimistic protocol as defined in Section 2.3.2.

The notation used during the explanation of the protocol is listed in Table 7.1 and the protocol message flow between the transacting parties are illustrated in Figure 7.1. We identify our main objectives of the protocol as follows.

1. The protocol should achieve strong fair-exchange while preserving anonymity of the consumer and the merchant.
2. TTP should not be able to see the exchanged product or store a copy of it.
3. Guarantee security properties such as; confidentiality, integrity, message freshness and non-repudiation.
4. Keep the involvement of the TTP to a minimum.
5. Disputes should be resolved within the protocol.
6. The protocol should support similar digital cash systems.

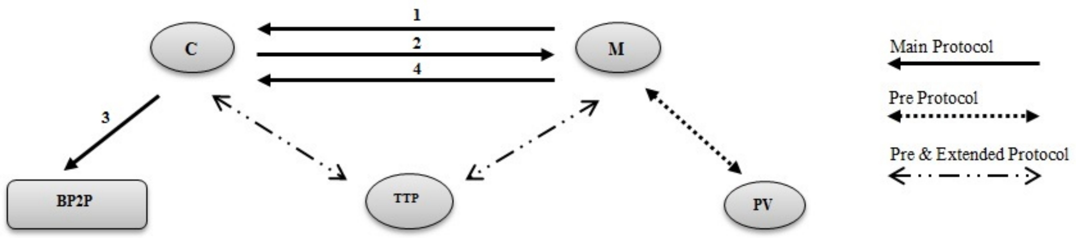


Figure 7.1: Proposed Protocol Message Flow.

There are three main phases of our proposed protocol. During the first stage of the protocol *"Pre Protocol"* a number of prerequisites such as product registration, product/price negotiation, TTP selection and product encryption are carried out. In the *"Main Protocol"* which is the second stage of the protocol, key processes such as product delivery, Bitcoin payment and decryption key delivery are carried out. If non of the processes fails and things go according to plan, the protocol completes in the above two stages. However, if a transacting party misbehaves, prematurely aborts or a communication failure happens the third stage *"Extended Protocol"* is executed with the involvement of the TTP. We make a number of assumptions listed in Section 7.3.1 for a successful run of the protocol.

Table 7.1: Notation used in the Proposed Protocols

C	:	Consumer.
M	:	The Merchant.
TTP	:	The Trusted Third Party.
PV	:	The Product Verifier.
$BP2P$	:	The Bitcoin Peer-to-Peer Network.
T_i	:	Purchase/delivery transaction of product m by C and M.
$Pseudo\text{-}ID\text{-}M$	:	Unique Pseudonym-Identity of M registered with the PV.
$Pseudo\text{-}ID\text{-}iX$	:	Unique Pseudonym-Identity of X registered with the TTP, only used during T_i.
K_1	:	Public encryption key of the public/private key pair escrowed with TTP, later used by PV to encrypt m.
K_1^{-1}	:	Private decryption key of the public/private key pair escrowed with TTP. The key pair is generated by M.
$eK_1\{Z\}$	:	Encryption of data string Z using a public algorithm with K_1.
P_X	:	Public Encryption Key of entity X.
$eP_X\{Z\}$	:	Encryption of data string Z using a public algorithm with the public encryption key P_X of entity X.
S_X	:	Private Signature Key of entity X.
$sS_X\,[Z]$	:	Digital signature outcome (without message recovery) from applying the private signature transformation on data string Z using the private signature key S_X of entity X.
V_X	:	Public Signature Verification Key of entity X.
P_{wM}	:	Public Encryption Key of M advertised online with m.
P_{iX}	:	Public Encryption Key of entity X used only during T_i.
$eP_{iX}\{Z\}$	:	Encryption of data string Z using a public algorithm with the public encryption key P_{iX} of X used only during T_i.
S_{iX}	:	Private Signature Key of entity X used only during T_i.
$sS_{iX}\,[Z]$	:	Digital signature outcome (without message recovery) from applying the private signature transformation on data string Z using S_{iX} of X only during T_i.
V_{iX}	:	Public Signature Verification Key of X only during T_i.
$V_{iX_{cert}}$	:	Public Signature Verification certificate issued by the TTP. It includes V_{iX} corresponding to the $Pseudo\text{-}ID\text{-}iX$ of entity X used only during T_i.
BP_X	:	Bitcoin Public Key of entity X (X's Bitcoin address).
BS_X	:	Bitcoin Private Key of entity X (X's Signature key).
$sBS_X\{Z\}$	:	Digital Signature outcome (without message recovery) from applying the private signature transformation on data string Z using BS_X of X.

$T\text{-}info$	:	Other information relevant to a particular Bitcoin transaction.
T_X	:	Bitcoin transaction from C to M, in the formation of a hash. $T_X = h(T_{(X-1)}\|\|BP_C\|\|T\text{-}info)$.
T_{X-1}	:	Previous Bitcoin transaction that has occurred in the past but directly linked to T_X in the formation of a hash.
$Encrypt_{cert}$	:	Encryption certificate issued by the PV. It includes a hash of the encrypted product which has been encrypted by the PV using the key indicated in the certificate.
TTP_{commit}	:	Commitment certificate issued by the TTP indicating involvement in the exchange.
$TTP\text{-}Pool$	:	A pool of different TTPs. C & M agrees between one TTP from this list to be involved in T_i.
PV_{cert}	:	Product verification certificate issued by the PV.
$\mathfrak{t}_{-payment}$	:	Predefined time-out for M to send the decryption key, includes time needed for Bitcoin transaction processing.
$\mathfrak{t}$	:	Predefined time-out period agreed by involved parties. If a response is not received within the time-out the sending party will resend once more, in case a no reply, the sending party aborts the protocol or involve the TTP if necessary.
$\mathfrak{t}_{-resolve}$	:	Time-out given to M by TTP to respond with the requested key before sending the escrowed product decryption key to C.
$A\|\|B$	:	Concatenation of A and B in that order.
$h(Z)$	:	Hash of data string Z.
N_{1X} / N_{2X}	:	First & second nonce issued by entity X.
$X \rightarrow Y : Z$	:	Entity X sends message Z to entity Y.

7.3.1 Assumptions in the Proposed Protocols

$A1$: M registers with PV by giving a $Pseudo\text{-}ID\text{-}M$, P_M and V_M which are only used in communication between M and PV. The PV is a trusted entity that certifies M's public verification key V_M and keep a record of it to verify M's messages signed using S_M in future communication.

$A2$: Both C and M register with TTP by giving a per-transaction pseudonym-identity $Pseudo\text{-}ID\text{-}iC$ and $Pseudo\text{-}ID\text{-}iM$. The TTP makes sure that each Pseudo-ID is unique and has not been registered before. It should not be possible for TTP, C, M or external parties to deduce the real identity of C and M by examining the Pseudo-ID.

$A3:$ C's and M's public verification keys V_{iC} and V_{iM} are certified by the TTP to their pseudonym-identities $Pseudo\text{-}ID\text{-}iC$ and $Pseudo\text{-}ID\text{-}iM$ respectively. The public certificates $V_{iC_{cert}}$ and $V_{iM_{cert}}$ are issued to each owner by TTP and can later be used to verify each other's digital signatures.

$A4:$ C and M have access to a Bitcoin wallet. M generates a one-time Bitcoin address to be presented to C to receive payments and only if needed, C generates a one-time Bitcoin address to receive any change back from the transaction also known as a change-address.

$A5:$ C and M, in addition to pseudonyms, maintain anonymity by setting up Anonymity Channels (uses cryptographic processes to change message origin details and prevent eavesdropping) for communication.

$A6:$ We also assume that the following guidelines are adhered to: all cryptographic keys are checked for validity before use, a standardised public key algorithm (e.g. RSA) is used for encryption, data is padded according to recommended best practice before encryption, hashes are generated using standardised secure hash functions (e.g. SHA) and messages are signed using a standardised digital signature algorithm (e.g. DSA).

$A7:$ The TTP is a trusted entity and will not collude with the PV, the M or the C.

$A8:$ The PV is a trusted entity, will not misrepresent the certified digital content and will not collude with either the TTP, the M or the C. The PV makes sure that the advertised product and the product description by M is exactly the same as that the product PV is certifying.

7.3.2 Pre Protocol Stage

The messages in the pre protocol stage are listed in Table 7.2 and described in detail below.

Firstly, the unique *product-ID* generated by M, *product-price*, *product-description* and a public key P_{wM} is advertised by the M online. The M can use his/her own website or a third party listing service to advertise these details. The TTP*-Pool* which is a list of potential TTPs that could be used in the transaction is advertised by the M. As per our assumption *A2*, a C who wish to purchase a product, selects a TTP from TTP*-Pool* and registers with it. The TTP selection process gives control to the C for choosing a TTP than relying on a particular TTP proposed by the M.

Table 7.2: Pre Protocol Stage

a.	$C \rightarrow M$	:	$encryption \| \| sS_{iC}[h(encryption)]$
			$encryption = eP_{wM}\{product\text{-}ID \| \| Order \| \| sS_{iC}[h(Order)] \| \| P_{iC} \| \| N_{1C} \| \| V_{iC_{cert}}\}$
			$Order = Pseudo\text{-}ID\text{-}iC \| \| TTP \| \| payment\text{-}method \| \| product\text{-}price$
b.	$M \rightarrow TTP$	:	$encryption \| \| sS_{iM}[h(encryption)]$
			$encryption = eP_{TTP}\{Transaction\text{-}ID \| \| K_1 \| \| K_1^{-1} \| \| Pseudo\text{-}ID\text{-}iM \| \| N_{1M}\}$
c.	$TTP \rightarrow M$	:	$encryption \| \| sS_{TTP}[h(encryption)]$
			$encryption = eP_{iM}\{K_1 \| \| TTP \| \| Pseudo\text{-}ID\text{-}iM \| \| sS_{TTP}[h(K_1 \| \| TTP \| \| Pseudo\text{-}ID\text{-}iM)] \| \| Transaction\text{-}ID \| \| N_{1M} \| \| N_{1TTP}\}$
d.	$M \rightarrow PV$	:	$encryption \| \| sS_M[h(encryption)]$
			$encryption = eP_{PV}\{pseudo\text{-}ID\text{-}M \| \| product\text{-}description \| \| m \| \| P_M \| \| product\text{-}ID \| \| N_{2M} \| \| K_1 \| \| TTP \| \| Pseudo\text{-}ID\text{-}iM \| \| sS_{TTP}[h(K_1 \| \| TTP \| \| Pseudo\text{-}ID\text{-}iM)]\}$
e.	$PV \rightarrow M$	:	$encryption \| \| sS_{PV}[h(encryption)]$
			$encryption = eP_M\{PV_{cert} \| \| Encrypt_{cert} \| \| N_{2M} \| \| N_{1PV}\}$
			$PV_{cert} = X_1 \| \| sS_{PV}[h(X_1)]$ $X_1 = product\text{-}ID \| \| product\text{-}description \| \| eK_1\{m\}$
			$Encrypt_{cert} = X_2 \| \| sS_{PV}[h(X_2)]$ $X_2 = h(eK_1\{m\}) \| \| K_1 \| \| TTP \| \| Pseudo\text{-}ID\text{-}iM$
f.	$M \rightarrow TTP$	:	$encryption \| \| sS_{iM}[h(encryption)]$
			$encryption = eP_{TTP}\{Transaction\text{-}ID \| \| Pseudo\text{-}ID\text{-}iM \| \| Pseudo\text{-}ID\text{-}iC \| \| Encrypt_{cert} \| \| N_{1TTP} \| \| N_{3M}\}$

g. $TTP \rightarrow M$: $\textit{encryption}||sS_{TTP}[h(\textit{encryption})]$

$\textit{encryption}=eP_{iM}\{TTP_{\textit{commit}}||N_{3M}||N_{2TTP}\}$

$TTP_{\textit{commit}}=Y_1||sS_{TTP}[h(Y_1)]$
$Y_1=Transaction\text{-}ID||Pseudo\text{-}ID\text{-}iM||$
$Pseudo\text{-}ID\text{-}iC||h(eK_1\{m\})$

Message a: The C registers with TTP and creates a concatenation which includes; *product-ID*, *Order*, a digital signature on the hash of *Order* using S_{iC}, P_{iC} only used in T_i, fresh nonce generated by C and the TTP issued public signature verification certificate.

The C then encrypts the concatenation using M's advertised public key. Then the hash of this encryption is signed by C using S_{iC} to create $sS_{iC}[h(\textit{encryption})]$. C sends both the digital signature and the encryption to M. We use the same notation to represent digital signatures sent in each subsequent message. The digital signatures can be verified by the M using $V_{iC_{cert}}$.

The *Order* includes; $Pseudo\text{-}ID\text{-}iC$ registered with TTP used only during T_i, TTP chosen and registered by C, *Payment-method* to indicate which digital cash system to use and *product-price*.

Message b: As detailed in our assumption *A2*, after receiving C's message, M registers with the same TTP. A concatenation is created by M which includes; a unique transaction ID generated by M for T_i, public and private key pair to be escrowed with TTP, $Pseudo\text{-}ID\text{-}iM$ registered with TTP used only during T_i and a fresh nonce.

M encrypts the concatenation using P_{TTP} and signs the hash of the encryption using S_{iM}. M then sends both parts to the TTP.

Message c: Once the message is received, TTP verifies whether the public/private key pair to be escrowed is in the correct format. If satisfied, TTP then creates a concatenation which includes; K_1 , TTP, $Pseudo\text{-}ID\text{-}iM$ (we refer to these as "the three components"), TTP's digital signature on the hash of the three components, the transaction ID, M's nonce and a new nonce. After this, TTP encrypts the concatenation using P_{iM} and signs the hash of the encryption using S_{TTP}. Both parts are then sent to the M.

Message d: After receiving TTP's message and registering with PV according to our assumption *A1*, M now needs to get product m certified and encrypted by the PV using K_1 escrowed with TTP. For this, M creates a concatenation which includes; *pseudo-ID-M*, *product-description*, m, P_M, *product-ID*, new nonce, the three components & TTP's digital signature on the hash of the three components. The concatenation is encrypted by M using P_{PV} and signs the hash of the encryption using S_M only used with PV according to *A1*. Both the encryption and the signature are then sent to PV.

Message e: PV after receiving the message, checks whether the product matches its product-description. If it matches, PV encrypts m using K_1 and generates a product verifier certificate PV_{cert} which includes X_1 and a signed hash of X_1 using S_{PV}. X_1 consists of the product−ID, product−description and encrypted product.

At the same time, PV also verifies TTP's digital signature on the three components received in the previous message. If satisfied, PV generates an *Encryption Certificate*. The $Encrypt_{cert}$ includes X_2 and a digital signature on the hash of X_2 using S_{PV}. X_2 consists of a hash of the encrypted product and the three components verified to have come from TTP. PV then creates a concatenation which includes; PV_{cert}, $Encrypt_{cert}$, N_{2M} and N_{1PV}. The concatenation is then encrypted using P_M which is shared only with PV. PV signs the hash of the encryption using S_{PV} and before sending both parts to M.

Message f: After receiving the message, M now creates a concatenation which includes; the *Transaction-ID*, *Pseudo-ID-iM*, *Pseudo-ID-iC*, $Encrypt_{cert}$, N_{1TTP} and a new nonce. The concatenation is then encrypted using P_{TTP} and the hash of the encryption is signed using S_{iM}. Both parts are then sent to TTP. It must be noted that, with the $Encrypt_{cert}$, the TTP only receives a hash of the encryption but not the actual encrypted product.

Message g: Lastly, once the message is received, the $Encrypt_{cert}$ is verified by the TTP. The verification indicates that the product was encrypted using key K_1 escrowed with TTP. Following this, a commitment certificate called that TTP_{commit} is issued by the TTP. The TTP then creates a concatenation which includes; the TTP_{commit}, M's previous nonce and N_{2TTP}. The concatenation is encrypted by P_{iM} and a signed hash of the encryption using S_{TTP} is appended before sending both parts to M. The TTP_{commit} includes Y_1 and a digital signature of TTP by signing the hash of Y_1 using S_{TTP}. Y_1 consists of the *Transaction-ID*, *Pseudo-ID-iM*, *Pseudo-ID-iC* and a hash of the encrypted product.

7.3.3 Main Protocol Stage

After successful completion of the pre protocol stage and examining the received TTP_{commit}, M initiates the main protocol. The messages communicated in the main protocol stage are listed in Table 7.3 and described in detail below.

Table 7.3: Main Protocol Messages.

1. $M \rightarrow C$	:	$encryption \| \| sS_{iM}[h(encryption)]$
		$encryption{=}eP_{iC}\{product\text{-}ID\|\|Invoice\|\|sS_{iM}[h(Invoice)]\|\| N_{1C}\|\|N_{4M}\|\|PV_{cert}\|\|TTP_{commit}\|\|V_{iM_{cert}}\|\|\mathfrak{t}\}$
		$Invoice{=}\{Transaction\text{-}ID\|\|product\text{-}price\|\|payment\text{-}method\|\| Pseudo\text{-}ID\text{-}iC\|\|Pseudo\text{-}ID\text{-}iM\|\|TTP\|\|BP_M\}$
2. $C \rightarrow M$	:	$encryption\|\|sS_{iC}[h(encryption)]$
		$encryption{=}eP_{iM}\{Invoice\|\|sS_{iM}[h(Invoice)]\|\| sS_{iC}[h(sS_{iM}[h(Invoice)])]\ \|\|N_{4M}\|\|N_{2C}\|\|h(eK_1\{m\})\|\|\mathfrak{t}_{-payment}\}$
3. $C \rightarrow BP2P$	:	$\{amount\|\|BP_C\|\|BP_M\|\|sBS_C[T_X\|\|BP_M]\ \|\|T_X\|\|T\text{-}info\}$
		$T_X = h(T_{(X-1)}\|\|BP_C\|\|T\text{-}info)$
4. $M \rightarrow C$	:	$encryption\|\|sS_{iM}[h(encryption)]$
		$encryption{=}eP_{iC}\{Invoice\|\|N_{2C}\|\|N_{5M}\|\|h(eK_1\{m\})\|\|K_1^{-1}\|\|\mathfrak{t}\}$

Message 1: Firstly, M creates a concatenation which includes; the *product-ID*, a newly created *Invoice*, a digital signature by signing the hash of the *Invoice* using S_{iM} to indicate that M agrees with the terms of the transaction, N_{1C}, new nonce, PV_{cert}, TTP_{commit}, TTP issued public signature verification certificate and a predefined time-out. The concatenation is encrypted using P_{iC} and a signed hash of the encryption using S_{iM} is appended. M then sends both parts to C. The *Invoice* consists of the *Transaction-ID*, *product-price*, *payment-method*, *Pseudo-ID-iC*, *Pseudo-ID-iM*, TTP and M's one-time Bitcoin address.

Message 2: After receiving the message, C decrypts it and retrieves $V_{iM_{cert}}$ to verify M's digital signature. The authenticities of PV_{cert} and TTP_{commit} is verified by C using PV's digital signature and TTP's digital signature respectively. The latter assures that TTP has confirmed involvement in the fair-exchange. Before continuing with the transaction process any further, C then carries out two main verification steps.

$$\begin{aligned}&Advertised\ Details = Details\ found\ in\ PV_{cert} \text{———————} ① \\ &Computed\ h(eK_1\{m\}) = h(eK_1\{m\})\ found\ in\ TTP_{commit} \text{———} ②\end{aligned}$$

Firstly, *product-ID* and *product-description* mentioned in the PV_{cert} is cross checked with the details of the product C is willing to purchase as advertised by M. If condition ① shown above is satisfied, then C is certain that the encrypted product m and its details as certified by the PV is the intended product that he/she is about to pay for.

Secondly, C generates a hash of the encrypted product $eK_1\{m\}$ and compares it with the hash obtained from TTP_{commit}. If both hashes match, it confirms that the hash of the encrypted product $eK_1\{m\}$ matches the hash TTP has confirmed to have the corresponding decryption key for. Then condition ② shown above is satisfied. This gives assurance to C that after making a payment, if M misbehaves, prematurely aborts or communication fails, the product decryption key can still be obtained by initiating the extended protocol. If and only if conditions ① & ② are satisfied, C proceeds to message 2 or else C aborts the protocol and informs both M & TTP the reasons.

Following this, C creates a concatenation which includes; $Invoice$, M's digital signature on the invoice, a digital signature by signing the hash of M's digital signature $sS_{iM}[h(Invoice)]$ using S_{iC} to indicate that C agrees with the terms of the transaction, N_{4M}, new nonce, hash of the encrypted product and a predefined time-out $\mathfrak{t}_{-payment}$. C then encrypts the concatenation using P_{iM} and appends a signed hash of the encryption using S_{iC} before sending both parts to M.

Message 3: Immediately after sending message 2, a Bitcoin payment to M's Bitcoin address BP_M is made by C. T_X is a hash output representing the Bitcoin transaction from C to M. The hash includes the previous transaction hash linking to this transaction, Bitcoin public key of C and other transaction information relevant to this transaction. Key components in the message broadcast to the $BP2P$ includes; the transferring amount, C's one-time Bitcoin public key, M's one-time Bitcoin public key, digital signature created by signing the transaction hash T_X using C's Bitcoin private signature key BS_C , T_X and other information related to the transaction.

The broadcast transaction is then received by the miners in the Bitcoin network, who then start creating a new block with the transaction information as follows. A detailed introduction to Bitcoin and the mining process was given in Section 2.2.2.

In brief, a Bitcoin miner computes a new block which includes the hash $T_{(X+1)}$ from this transaction. In $BP2P$ every miner engages in computing blocks simultaneously. Due to this reason only the first valid block created is verified by other peers to be genuine and a new record is added to the Blockchain. The Bitcoin network at the same time checks whether the Bitcoins have been spent previously to detect double spending.

Message 4: M waits for C's Bitcoin payment to be confirmed in the Blockchain. This process and the time required was discussed in Section 2.2.2. After receiving the payment, M now needs to send the product decryption key K_1^{-1} to C. For this M creates a concatenation which includes; the $Invoice$, N_{2C}, a new nonce, $h(eK_1\{m\})$, K_1^{-1} and a time-out. M then encrypts the concatenation using P_{iC} and appends a signed hash of the encryption using S_{iM} before sending both parts to the C.

If the main protocol messages complete successfully then C decrypts the product and M updates $Transaction\text{-}ID$ as completed in his/her record.

7.3.4 Extended Protocol Stage

The extended protocol is executed in the event that M misbehaves by sending an incorrect decryption key, prematurely aborts or a communication failure happens after C making a Bitcoin payment. The protocol messages are listed in Table 7.4 and described below.

Table 7.4: Extended Protocol Messages.

I. $C \rightarrow TTP$	:	$encryption \| \| sS_{iC}[h(encryption)]$
		$encryption{=}eP_{TTP}\{BlockchainEvidence\|\|Invoice\|\| sS_{iM}[h(Invoice)]\|\|N_{3C}\|\|h(eK_1\{m\})\|\|TTP_{commit}\|\|\mathfrak{t}\}$
II. $TTP \rightarrow M$	:	$encryption\|\|sS_{TTP}[h(encryption)]$
		$encryption{=}eP_{iM}\{BlockchainEvidence\|\|Invoice\|\| sS_{iM}[h(Invoice)]\|\|KeyRequest\|\|N_{3TTP}\|\|\mathfrak{t}_{-resolve}\}$
III. $M \rightarrow TTP$	:	$encryption\|\|sS_{iM}[h(encryption)]$
		$encryption{=}eP_{TTP}\{Invoice\|\|K_1^{-1}\|\|N_{3TTP}\|\|N_{6M}\|\|\mathfrak{t}\}$
IV. $TTP \rightarrow C$	:	$encryption\|\|sS_{TTP}[h(encryption)]$
		$encryption{=}eP_{iC}\{Invoice\|\|K_1^{-1}\|\|N_{3C}\|\|N_{4TTP}\|\|\mathfrak{t}\}$

Message I: C initiates the protocol by creating a concatenation which includes; evidence from the blockchain showing the Bitcoin payment from the C to M, $Invoice$, M's digital signature on the hash of $Invoice$ received in message 1, a new nonce, hash of the encrypted product generated by C, TTP_{commit} and a predefined time-out. C encrypts the message using P_{TTP} and signs the hash of the encryption using S_{iC} before sending both parts to the TTP.

Message II: TTP examines the evidence produced by C in message I. The TTP_{commit} confirms that TTP is involved in the particular transaction and $sS_{iM}[h(Invoice)]$ confirms that M has agreed to the terms of T_i. The $BlockchainEvidence$ is verified by TTP by examining the Bitcoin blockchain which should confirm the Bitcoin payment made from C to M. As a response to this, TTP creates a concatenation which includes; the $BlockchainEvidence$, $Invoice$, $sS_{iM}[h(Invoice)]$, a $KeyRequest$ from the TTP requesting for K_1^{-1}, a new nonce and a predefined time-out $\mathfrak{t}_{-resolve}$. TTP encrypts the concatenation using P_{iM} and appends a signed hash of the encryption using S_{TTP} before sending both parts to M.

Message III: If M has not maliciously disappeared after receiving C's payment, misbehaved or deliberately refused to communicate, then as soon as TTP's key request is received, M checks the status of the $Transaction\text{-}ID$. If it has not completed, M creates a concatenation which includes; $Invoice$, the product decryption key, N_{3TTP}, a new nonce and a predefined time-out. M encrypts the concatenation using P_{TTP} and signs the hash of the encryption using S_{iM} before sending it to TTP.

Message IV: TTP proceeds to message IV as normal if a response to the key request is received within $\mathfrak{t}_{-resolve}$. However, if M's response is not received then TTP retrieves the product decryption key escrowed with itself and creates message IV. In either scenario, TTP creates a concatenation which includes; $Invoice$, K_1^{-1}, N_{3C}, a new nonce and a time-out. The concatenation is then encrypted using P_{iC} and a signed hash of the encryption using S_{TTP} is appended before both parts are sent to the C.

After receiving the message, C retrieves the decryption key which gives access for the C to successfully decrypt the purchased product.

7.4 Security Analysis

In this section, we analyse our proposed protocol for its security requirements and to see whether it achieves the objectives outlined in Section 7.3.

7.4.1 Strong Fair-exchange

We start our analysis by evaluating the fairness guarantees provided by our proposed protocol. It must be noted that, if a party aborts the proposed protocol before C makes the Bitcoin payment in *Message 3*, fair-exchange is not affected as neither party gains an advantage from the other. This means that M will not be made a Bitcoin payment and C will not be able to decrypt the product.

In the particular transaction scenario where both C and M followed the protocol without misbehaving or prematurely aborting, the protocol completes without an extended protocol stage. In our analysis we consider the following transaction scenarios and evaluate the fairness of our proposal in such instances.

- **M sends a wrong encrypted product:** M gains no advantage by sending a wrong product as C only proceeds to making a payment once conditions ① & ② are satisfied. C can detect any potential wrong products at this stage.
- **M sends a wrong product decryption key:** In this case, C initiates the extended protocol by sending all relevant evidences to TTP as shown in *Message I* to raise a dispute resolution. At the end of the extended protocol TTP forwards C the correct decryption key.
- **M after receiving payment demands more payment:** In this instance, C initiates the extended protocol. By looking in to evidence C has provided in *Message I*, TTP would determine that M has agreed to the terms of the transaction and the price by digitally signing the $Invoice$.
- **M disappears after receiving payment or aborts:** In such scenario, C initiates the extended protocol to involve TTP for dispute resolution.
- **C pays *M* less than the agreed amount:** In this scenario, the decryption key to the encrypted product is only sent to C if the full payment is received by M. Due to this C doesn't gain any advantage by making partial payments. If M aborts the protocol at this stage where C has paid less then the agreed amount, both parties have misbehaved. Therefore, both parties have failed to follow the protocol and there is no fair-exchange to achieve. If C wants to raise a dispute resolution with the TTP, then C needs to make the full payment.
- **C pays *M* less and initiates the extended protocol:** If C claims deceivingly that the amount paid is what he/she agreed, then M could provide evidence upon

enquiry by TTP in *Message III*. In this scenario, M provides the signed purchase order $sS_{iC}[h(Order)]$ in *Message c*, and if received $sS_{iC}[h(sS_{iM}[h(Invoice)])]$ in *Message 2*, which both includes C's digital signature agreeing to the terms of the transaction. Therefore, the decryption key is not provided to the C.

- **C collude with *TTP*:** In *A7* we assumed that the trusted TTP is a trusted entity and will not collude with the C. Therefore, this concern is not realised within our protocol assumptions. However, we discuss below the outcome if the potential concern is realised. Since the choice of selecting a TTP from the $TTP-Pool$ is given to C, it is more likely for C to collude with TTP than M. This may disadvantage M as TTP could send C the escrowed decryption key before C's Bitcoin payment to M. However, anonymity of M cannot be breached this way as M never reveals the real identity to any of the parties involved in the protocol.

- **M collude with *TTP*:** Similarly according to *A7*, this potential concern is not realised within our protocol assumption. However, we discuss below the outcome if the potential concern is realised. There is less chance for this to be realised as the choice of selecting a TTP is given to the C. In such event, this may disadvantage C as TTP could send C an incorrect decryption key in the extended stage after C's Bitcoin payment to M. However, anonymity of C is not breached.

- **C and *M* collude together:** As both parties are dishonest there is no fair-exchange to be achieved.

In our analysis above, it is evident that the protocol guarantees that in the event of a transacting party misbehaving, the genuine party who follows the protocol doesn't incur any loss or have to go through a lengthy dispute resolution process with an external judge after the protocol. Instead, in our proposed protocol disputes are resolved within the protocol run. All these properties constitute strong fair-exchange which is established by our proposed protocol.

7.4.2 Anonymity

It must be noted that both C and M do not reveal their real identities or personal details at any stage of the proposed protocol. Instead, per-transaction pseudo-IDs and public/private key pairs are used by both parties. During the protocol, C communicates with M and TTP using *pseudo-ID-iC* only used in T_i and does not communicate with

PV. M communicates with C and TTP using *Pseudo-ID-iM* only used in T_i and with PV using *pseudo-ID-M* for product registrations. Because of this reason anonymity of C and M is not only guaranteed between each other but also to PV and TTP. Attempts of collusion with PV or TTP to find the real identity of a party, would not gain any benefit other than what they already know. e.g. C colluding with TTP to identify M.

In addition to pseudonyms, according to our assumption in *A5*, both C and M set-up anonymity channels for communication between each other and other parties. This provides assurance that the communication channel that the protocol runs on does not reveal any information related to the identities of the communicating parties. It is common practise to use an anonymiser such as the *TOR Browser* while making Bitcoin payments.

Furthermore, the payment made by C is sent to a one-time Bitcoin address generated by M. Similarly when making the payment, C generates a one-time Bitcoin address to receive Bitcoins as change only if there are any. Another aspect to point out is that any information related to Bitcoin transactions and addresses are not revealed to the TTP unless the extended protocol is initiated.

Bitcoin addresses does not reveal the real identity of the user. However, as the Bitcoin blockchain is a public ledger, all Bitcoin transactions are broadcast publicly, which makes Bitcoin transactions completely transparent. This feature can be exploited in attempts to de-anonymise Bitcoin and to reveal the real identities of Bitcoin users. For an example, with the advancements in computational power, machine learning and *Data Analysis* capabilities, it may be possible to link Bitcoin transactions to real user identities [164, 137]. It must be noted that, none of these attempts has fully broken the anonymity provided by the Bitcoin protocol. However, this raises a few concerns over the anonymity of the users and the privacy of transactions in our proposed protocol. Therefore, in Section 7.5, we add an extension to our protocol to support Zerocoin/Zerocash system proposed in [141, 48] to provide improved privacy grantees and full anonymity.

7.4.3 Privacy of Exchanged Product

In our protocol, neither product m is revealed nor given a copy to the TTP. Any information related to m is only revealed to the TTP after the execution of the extended protocol. Even then the details included in *Invoice* are not sufficient for TTP to find out what was exchanged.

7.4.4 Security Properties

Confidentiality is guaranteed by using strong encryption on all protocol messages and the use of nonces confirms message freshness. The unforgeability of the digital signature algorithm and the use of registered pseudonym-IDs provide non-repudiation. This assures that a party cannot deny taking part in the protocol at a later stage. Message integrity is provided by the digital signatures appended on each protocol message. The predefined time-outs provide timeliness to the proposed protocol. This allows the sender to resend a message once more if a response is not received, to complete the protocol by aborting or to complete the protocol by resolving with the TTP without letting the protocol go in to an infinite loop.

7.4.5 Other Properties

- **Minimum involvement of $\boldsymbol{TTP}$**: Our proposed protocol uses an off-line TTP by keeping its intervention to a minimum. The TTP is only involved with the protocol if a transacting party misbehaves, prematurely aborts or a communication failure happens. This makes our proposed solution an optimistic protocol as defined in Section 2.3.2.
- **Dispute resolution**: In our proposed protocol, any disputes related to the payment or exchange of digital content is resolved automatically within the protocol run without manual intervention of an external judge or going through a long dispute resolution process after the protocol run.

7.4.6 Support for Other Cryptocurrencies

In our proposed solution, we gave more emphasis on Bitcoin mainly because of the identified problems in Section 7.2. Further more, other aspects such as the following were also taken into consideration: at the time of writing, Bitcoin was the most popular, widely used and the cryptocurrency that had the most market capitalisation. Even though Bitcoin was selected as the payment method for the protocol, the protocol can be extended to support other anonymous cryptocurrencies that are designed using a public ledger/blockchain technology.

We now describe a generalised example of how the protocol can be extended to support other cryptocurrencies designed with the concept of providing anonymity using pseudonymous addresses and a public ledger.

In *Message a* of the proposed protocol, C can inform M of the currency that he/she would like to use in the *payment-method*. Following this, M can later send the relevant address of the selected payment method in *Message 1*. C then makes a payment using the selected payment system. If things go according to plan and M sends the decryption key then the protocol completes successfully.

However, in the event of a party misbehaving, prematurely aborting the protocol or in a communication failure, the extended protocol can be executed. Then C could present evidence from the relevant cryptocurrency's public ledger in *Message I*. This evidence may include payment made from payer's address to the address of the payee available on the public ledger. This provides sufficient information for TTP to continue with the dispute resolution without requiring further changes to the protocol.

In Section 7.5 we describe this further and extend our protocol to support Zerocoin/Zerocash. There are a few differences in our extended protocol compared to the generalised example. This is mainly because of how Zerocoin/Zerocash works to provide increased anonymity as well as privacy of transaction details published in the blockchain.

7.5 Extension to Support Zerocoin/Zerocash

The notation used in our extended protocol is listed in Table 7.5. To make cross-referring easy for the reader, the notation we present is similar to the notation in corresponding papers [48, 141].

Table 7.5: Notation used for Zerocoin/Zerocash Extension

$ZP2P$	:	Zerocash System integrated into Bitcoin P2P Network.
PRF	:	Pseudorandom function.
$COMM$	:	Statistically-hiding non-intractive commitment scheme.
$zk\text{-}SNARK$	:	zero-knowledge Succinct Non-interactive ARguments of Knowledge.
rt	:	A root of Merkle tree at a given time.
a_{pk_x}	:	Address Public Key of entity X (X's Zerocoin address).
a_{sk_x}	:	Address Secret Key of entity X.
z_x	:	A Zerocoin that is owned by entity X.
z_x^{old}	:	A Zerocoin that is owned by entity X and is used to pour it's value to new coins.
v_x	:	The value of entity X's Zerocoin.
v_{pub}	:	A non-negative public output value that can be used to pay a target similar to a Bitcoin address as specified in a transaction string $info$.

ρ_x	:	A secret value that determines sn_x of entity X's coin.
sn_x	:	A serial number derived as $sn_x = PRF^{sn}_{a_{sk_x}}(\rho_x)$.
cm_x	:	A coin commitment of entity X's coin (a string that appears in the public ledger) constructed as $k_x = COMM_{r_x}(a_{pk_x}\|\|\rho_x)$ and $cm_x = COMM_{s_x}(v_x\|\|k_x)$ where r_x & s_x are random.
tx_{Mint}	:	A mint transaction record; when a new coin z with commitment cm and value v has been minted.
π_{POUR}	:	A *zk-SNARK* proof that states; "Given rt, old sn_x, new commitments cm_x and cm_y, I know coins z^{old}, new coins z_x, z_y and old address secret key a_{sk_x}".
tx_{Pour}	:	A pour transaction is used to spend, split, merge or transfer ownership of anonymous coins to others. Pour records the pouring of a old coin/two coins (z^{old}_x) with their corresponding serial numbers (sn^{old}_x) into two new coins (z_x, z_y) with their commitments (cm_x, cm_y) in the public ledger. It also records rt, v_{pub}, $info$ and π_{POUR}.

In this section, we only present the changes in the protocol run that are required to extend our protocol to support Zerocoin/Zerocash system. The modified protocol messages are listed in Table 7.6 and described in detail below. The changes to our previous protocol are shown in highlighted text.

Message f: At this protocol stage, M generates a new address key pair (a_{pk_m}, a_{sk_m}) and a secret value ρ_m in order for C to make a Zerocoin payment. After this, M constructs a coin commitment cm_m as:

$k_m = COMM_{r_m}(a_{pk_m}||\rho_m)$ and $cm_m = COMM_{s_m}(v_m||k_m)$ where r_m & s_m are random. The coin commitment cm_m is then included in *Message f* by M for TTP's record.

Message g: After receiving the message, TTP keeps a record of cm_m and includes the cm_m in the commitment certificate TTP_{commit} generated by the TTP. By including the cm_m in TTP_{commit} gives assurance to C that TTP is aware of the corresponding commitment of M's coin that is due to appear in the public ledger. The TTP_{commit} is sent to C in *Messages* 1. C can also check the validity of cm_m by reconstructing it using secret values $a_{pk_m}, v_m, \rho_m, r_m, s_m$ received in *Messages* 1.

Message 1: In the extended protocol, instead of including the Bitcoin address, M now includes address public key a_{pk_m} in the *Invoice*. Furthermore, secret values v_m, ρ_m, r_m, s_m are also included in the *Invoice* to provide required information for C to make an anonymous payment to M.

In our protocol, we let M generate these details instead of C. There are two main reasons for this approach. These are:

Table 7.6: Protocol Extension to Zerocoin/Zerocash.

f. $M \rightarrow TTP$	:	$encryption \| sS_{iM}[h(encryption)]$ $encryption{=}eP_{TTP}\{Transaction\text{-}ID \| Pseudo\text{-}ID\text{-}iM \|$ $Pseudo\text{-}ID\text{-}iC \| Encrypt_{cert} \| N_{1TTP} \| N_{3M} \| \mathbf{cm_m}\}$
g. $TTP \rightarrow M$	:	$TTP_{commit}{=}Y_1 \| sS_{TTP}[h(Y_1)]$ $Y_1{=}Transaction\text{-}ID \| Pseudo\text{-}ID\text{-}iM \| Pseudo\text{-}ID\text{-}iC \|$ $h(eK_1\{m\}) \| \mathbf{cm_m}$
1. $M \rightarrow C$	:	$Invoice{=}\{Transaction\text{-}ID \| product\text{-}price \|$ $payment\text{-}method \| Pseudo\text{-}ID\text{-}iC \| Pseudo\text{-}ID\text{-}iM \|$ $TTP \| \mathbf{a_{pk_m}} \| \mathbf{v_m} \| \rho_\mathbf{m} \| \mathbf{r_m} \| \mathbf{s_m}\}$
3. $C \rightarrow ZP2P$	:	$\mathbf{z_m} = (\mathbf{a_{pk_m}} \| \mathbf{v_m} \| \rho_\mathbf{m} \| \mathbf{r_m} \| \mathbf{s_m} \| \mathbf{cm_m})$ $\mathbf{z_c} = (\mathbf{a_{pk_c}} \| \mathbf{v_c} \| \rho_\mathbf{c} \| \mathbf{r_c} \| \mathbf{s_c} \| \mathbf{cm_c})$ $\mathbf{tx_{Pour}} = (\mathbf{rt} \| \mathbf{sn^{old}} \| \mathbf{cm_m} \| \mathbf{cm_c} \| \pi_\mathbf{POUR} \| \mathbf{v_{pub}} \| \mathbf{info})$
I. $C \rightarrow TTP$	:	$encryption \| sS_{iC}[h(encryption)]$ $encryption{=}eP_{TTP}\{\mathbf{cm_m} \| \mathbf{Invoice} \| sS_{iM}[h(Invoice)] \|$ $N_{3C} \| h(eK_1\{m\}) \| TTP_{commit} \| \mathfrak{t}\}$
II. $TTP \rightarrow M$	:	$encryption \| sS_{TTP}[h(encryption)]$ $encryption{=}eP_{iM}\{\mathbf{cm_m} \| Invoice \| sS_{iM}[h(Invoice)] \|$ $KeyRequest \| N_{3TTP} \| \mathfrak{t}_{-resolve}\}$

1. To keep the protocol simple by not requiring a *key-private encryption scheme* to download these secret values in encrypted format from the public ledger as specified in the paper [48]. The main reason for this is that we have already established a secure channel in our protocol.

2. To make M generate the coin commitment and get it added to TTP_{commit} before C makes a payment.

Message 3: Immediately after sending *Messages* 2, a Zerocoin payment using the Zerocash system is made by C. Assuming C is a Bitcoin user, a new Zerocoin can be minted by depositing a Bitcoin with a backing escrow pool in the Zerocash system. If C is already a user that holds Zerocoins then the previous step of minting a coin is not needed.

Because C uses this newly minted coin to make new coins, we add the notation *old* to it's parameters. Firstly, C generates a new address key pair $(a_{pk_c}^{old}, a_{pk_c}^{old})$ and a secret value ρ_c^{old} which determines coin z_c^{old}'s serial number $sn_c^{old} = PRF_{a_{sk_c}^{old}}^{sn}(\rho_c^{old})$. It is assumed that these serial numbers are collision resistant.

C now generates cm_c^{old} as:

$k_c^{old} = COMM_{r_c^{old}}(a_{pk_c}^{old}||\rho_c^{old})$ and $cm_c^{old} = COMM_{s_c^{old}}(v_c^{old}||k_c^{old})$ where r_c^{old} & s_c^{old} are random.

The minting outputs a new coin and a mint transaction:

$z_c^{old} = (a_{pk_c}^{old}||v_c^{old}||\rho_c^{old}||r_c^{old}||s_c^{old}||cm_c^{old})$

$tx_{Mint} = (v_c^{old}||k_c^{old}||s_c^{old}||cm_c^{old})$

In the Zerocash system, to spend the newly created coin, C carries out a *pour* operation which takes z_c^{old} as input coin and pours it's value into two fresh coins; z_m & z_c.

C then uses z_m to make M's payment and z_c to pay any *change back* from the transaction to him/her-self. To create these two coins C firstly, generates commitment cm_m for M's coin using the secret values received in *Messages* 1, such that; $k_m = COMM_{r_m}(a_{pk_m}||\rho_m)$ and $cm_m = COMM_{s_m}(v_m||k_m)$.

C also at this point checks whether the constructed commitment matches the one found in TTP_{commit}. C now generates the commitment for his/her own new coin such that; $k_c = COMM_{r_c}(a_{pk_c}||\rho_c)$ and $cm_c = COMM_{s_c}(v_c||k_c)$ where r_c & s_c are random. Following this, C produces a π_{POUR} proof according to [48] and the serial number $sn_c^{old} = PRF_{a_{sk_c}^{old}}^{sn}(\rho_c^{old})$.

The *pour* operation outputs two new coins and a tx_{Pour} that is appended to the public ledger.

$z_m = (a_{pk_m}||v_m||\rho_m||r_m||s_m||cm_m)$

$z_c = (a_{pk_c}||v_c||\rho_c||r_c||s_c||cm_c)$

$tx_{Pour} = (rt||sn^{old}||cm_m||cm_c||\pi_{POUR}||v_{pub}||info)$

M who is expecting a Zerocoin payment from C can now start using the received coin without having to scan the entire public ledger using the *Receive* algorithm beforehand as specified in [48]. This is possible because the cm_m and the private values was communicated to the M before receiving the payment.

After receiving the payment, M can now pour the value of the received coin to a new coin owned by M using the *pour* operation. This makes the parameters related to the coin such as the coin commitment only known to M. Furthermore the value can be transferred to another owner in a similar way how C transferred the value to M.

Messages I: If the payment is received successfully and the decryption key is sent in *Messages* 4, then the protocol completes without going to an extended stage. However, after C making a payment, if M misbehaves by sending an incorrect decryption key, prematurely aborts or a communication failure happens then the extended protocol can be initiated. Here, C appends cm_m and the new $Invoice$ instead of $BlockchainEvidence$ in *Messages I*. It must be noted that the $Invoice$ is only revealed to the TTP if the extended protocol is initiated.

Messages II: After receiving C's message, TTP verifies the cm_m to see whether it matches the one in TTP_{commit}.

Furthermore, TTP checks whether the cm_m has appeared in the public ledger already. For this task, TTP can use the secret values (v_m, ρ_m, r_m, s_m) found in the $Invoice$ to query the blockchain.

If the cm_m has been recorded in the blockchain, this gives assurance to TTP that a payment was made to M's Zerocoin address corresponding to the coin's commitment cm_m. If satisfied, TTP contacts the M as the next step in the dispute resolution by sending the $KeyRequest$ message which also includes cm_m. In either scenario, whether M forwards the product decryption key or not within the predefined time-out, TTP retrieves the escrowed decryption key and forwards it to C.

7.5.1 Security and Anonymity

Our previous protocol was aimed at achieving strong fair-exchange while using Bitcoin as the payment method. However, as identified in our analysis in Section 7.4, recent work has shown that it may be possible to link Bitcoin transactions to real identities [164, 137]. Addressing these concerns, our extended protocol supports Zerocash as a payment method while providing improved transaction anonymity for users.

The protocol objectives as discussed in Section 7.4 are also achieved in our extended protocol, even though a separate analysis is not mentioned here. In the extended protocol, the corresponding payment transaction from C to M is not publicly available in the ledger as in a Bitcoin transaction. It should also be noted that despite the fact that C & TTP get to know M's secret values (v_m, ρ_m, r_m, s_m) and commitment cm_m, coin z_m cannot be spent by either of these two parties as address secret key a_{sk_m} is only known by M. Furthermore, the output serial number $sn_m = PRF^{sn}_{a_{sk_m}}(\rho_m)$ is not revealed at any stage of our protocol and because of this when M spends z_m, it still cannot be traced which provides additional anonymity to the transaction.

7.5.2 Mechanical Formal Analysis

The main protocol stage proposed in Table 7.3 and the extended protocol stage proposed in Table 7.4 were subject to mechanical formal analysis using Scyther [80]. The main reason for choosing the main and the extended protocol stages for mechanical formal analysis was because, in the fair-exchange between the merchant and the customer, these phases of the protocol are critical for guaranteeing the security requirements. It includes security sensitive stages such as: delivery of the encrypted digital content and delivery of the decryption key to the customer. Therefore, providing the required security levels at these stages is important.

The protocol was analysed using the Dolev-Yao adversarial model [87]. The following security claims were successfully verified during the protocol analysis: Aliveness *(Alive)*, Weak agreement *(Weakagree)*, Non-injective agreement *(Niagree)* Non-injective synchronisation *(Nisynch)* of the transacting entities and Secrecy of data *(Secret)* for: m and K [81, 80]. In addition to the claim types defined above, the *verify automatic claims* feature on Scyther was used to verify other claims [80].

Following successful execution of the script, the security of data in the claim events were verified and Scyther did not find any feasible attacks. The Scyther script is available in Appendix A.4 and can be downloaded from [7] and [6].

7.6 Summary

The chapter identified a problem associated with established fair-exchange while using anonymous payment such as Bitcoin in e-commerce environments. We then investigated background work related to fair-exchange and anonymity. Following this, we proposed a protocol that achieves strong fair-exchange while preserving anonymity of the transacting parties. In the protocol the involvement of the TTP is kept to a minimum by using an off-line TTP that only intervenes if something goes wrong. Any dispute that arise during the protocol is resolved within the protocol run.

Furthermore, the TTP agreed between C & M does not get to see the exchanged product or store a copy of it. We also present an extended protocol that gives support of using other cryptocurrencies based on public ledgers/blockchains. Finally we outline a concern in Bitcoin that raises anonymity concerns and propose an extension to support Zerocash which addresses this issue while providing improved transaction anonymity for users.

Chapter 8

Blockchain based Philanthropic Model and Payment System

Contents

In this chapter we focus our attention on exploring other payment scenarios that can be enhanced by leveraging blockchain technology. We identify the philanthropic sector as an industry that has huge potential to improve donation payments associated with humanitarian aid activities using blockchain technology. In our research we identify challenges faced by charities and propose a novel philanthropic model for donating foreign aid using the Bitcoin blockchain as an example. We then propose a SMS based mobile payment solution to provision donations and to be used by beneficiaries in disconnected environments. The solution is finally analysed for its security requirements.

8.1 Introduction

The blockchain technology is one of the main innovations that came out of Bitcoin. Since its introduction, blockchain technology is gaining rapid interest due to its distributed nature and strong security properties [202, 199]. However, the use of blockchains is not limited to constructing decentralised cryptocurrencies, but also can be applied to other innovative ideas such as smart contracts, recording asset ownerships, cross-border payment solutions, trade finance, etc. [199, 196].

In this chapter we explore how the blockchain technology can be leveraged to provide services in the philanthropic sector, which provides invaluable social and humanitarian services. A report by the UK Charities Aid Foundation [67] identifies that for charities, blockchain technology can increase transparency, openness and trust whilst reducing transaction costs and providing new opportunities for fundraising. The sector is very large: according to the Charities Aid Foundation (CAF) - World Giving Index 2015 report, around 1.4 Billion people donated money in that year [66]. The findings were based on an ongoing research project that represents around 96% of the world population in their study [66]. Furthermore, according to the Charity Commission Annual Report and Accounts 2015-16 (United Kingdom), the Charity Commission regulated £70.93 Billion charity income [184], donated by governments, businesses and individuals.

In the United Kingdom in 2015, the Charity Commission regulated £70.93 Billion charity income [184], donated by governments, businesses and individuals. However, public trust in charities is declining [108, 157, 75, 76, 197]. Former U.N. Secretary-General Ban Ki-moon's closing remarks at high-level panel on accountability, transparency and sustainable development was "Last year, corruption prevented 30 per cent of all development assistance from reaching its final destination. This is a failure of accountability and transparency" [192]. Negative media stories about charity management (such as the investigation into mismanagement at Kids' Company [186, 75]); and general worries about how charities collect and spend donations have impacted on the confidence that the general public has in the charity sector [157]. As a result there is a groundswell of opinion that charities need to do more to improve their accountability and transparency to donors: these aspects could be enhanced greatly if charities used blockchain technology in their donor transactions, because transparency is inherent in a distributed ledger approach [156, 86, 108, 67].

There are existing cases where donations in cryptocurrencies have provided charities with advantages. For example, the Royal National Lifeboat Institution (RNLI) in the UK has accepted Bitcoin since August 2015, hoping to attract new donors from a different demographic to its typical supporters [51]. Another example is a donation tracking service (called GiveTrack) set up by the BitGive Foundation, that allows donors and third parties to trace transactions in real time: this shows how donations are spent, and provides some trust that the funds reach their intended destination [29]. This is built on the fact that the blockchain is a public ledger available globally: as every donation is associated with a Bitcoin address, therefore anybody with this address can easily obtain a transparent audit trail for a particular donation.

The chapter identifies challenges that charities and Non Profit Organisations (NPOs) face which could be addressed by the use of blockchain technology. These are: donation transparency, uniqueness, and provisioning; and reducing transaction costs. To this end, we introduce a blockchain based philanthropic model, that utilises a web-based donation platform where donations can be made either using Bitcoin or fiat currency[1].

This donation platform allows individual and corporate donors to choose which areas of the charity's operations they wish to support. As Bitcoin transactions are public and unique, this provides audit trails and feedback about how each individual donation was used, improving transparency and trust in the charity. The back-end payment transactions are carried out using multi-signature Bitcoin payments to enhance security. More advanced services can be offered using Smart Contracts via the Rootstock platform [4]. Furthermore, it allows the charity to provide feedback about how each individual donation was used. This model provides the charity with the advantages of speedy transactions at reduced cost, donation transparency (and hence higher trust in the charity) and an infrastructure that can be used for donation provisioning.

This generic philanthropic model is then used in a case study of foreign aid distribution in a geographical environment with poor Internet infrastructure, lack of supported devices and unbanked [2] beneficiaries. Here the priority is to get financial aid to beneficiaries efficiently and securely, whilst minimising the potential for fraudulent transactions. This case study illustrates how the proposed philanthropic model could work even when the basic technological infrastructure necessary for a blockchain solution (i.e. Internet connectivity and supported devices) is not available.

Examples of such disconnected environments include warzones, disaster areas, or economically deprived areas of the world. There are many operational and security problems in distributing financial aid in these challenging conditions as conventional internet-based money transfer may not be possible and physical cash handling may be fraught with danger [64]. In the proposed scheme, the charity will maintain hosted

[1]Fiat currency is a currency declared by government regulation as legal [130].
[2]Unbanked means not having access to a bank or financial institution [190].

Bitcoin Wallets for charity workers, beneficiaries and merchants: Bitcoin transactions can be done using low-cost security tokens (distributed per-user), basic mobile phones and an SMS mobile payment system utilising an existing GSM network.

The main contributions of the chapter are: 1) a new philanthropic model that leverages the Bitcoin blockchain for charitable donations / donation provisioning and 2) an SMS based Bitcoin mobile payment system that can be used in an offline environment.

The chapter is structured as follows. Benefits of blockchains for a charity/NPO are identified in Section 8.2 and a new philanthropic model that addressed these challenges is introduced in Section 8.3. Constraints of blockchain solutions are discussed in Section 8.4. In Section 8.5, a use case of applying the proposed philanthropic model to provide humanitarian aid in a disconnected environment is discussed. The proposed scheme is evaluated in Section 8.6. Finally in Section 8.7, the discussion is concluded..

8.2 Benefits of Blockchain Solutions for Charities/NPOs and Donors

In this section, useful features that blockchain technology can bring to charities/NPOs and donors are discussed. An introduction to Blockchain technology and Bitcoin was given in Sections 2.2.1 and 2.2.2.

Charities need to do more to improve their accountability and transparency to donors: these aspects could be enhanced greatly if charities used blockchain technology in their donor transactions, because transparency is inherent in the distributed ledger approach [86, 108, 67, 51, 29]. A blockchain solution would provide the following benefits for charities and donors:

1. Donation transparency: We define this as the publicly available audit trail of a particular donation made by an individual donor that details exactly where the donation went and whether it reached the intended charitable activity. Every donation is associated with a Bitcoin address, therefore anybody with this address can easily obtain its publicly available audit trail detailing exactly where a particular donation went. Currently, there is no mechanism for a donor to obtain an audit trail of their donation. Providing donation transparency would help enhance the donor's donation experience and the trust that he/she has in the charity.

2. Reducing transaction costs: A global charity's operational costs may involve costly international transaction fees and interchange[3] that may have to be passed on to

[3]Interchange fee is a cost that is paid by the acquirer to the card issuing bank. The transaction value is paid to the merchant after deducting the interchange [187].

the donor: donors may also incur additional international transaction fees from their payment provider. This could negatively affect a potential donor's willingness to make donations. The global average cost for sending remittances worldwide was 7.42% in the third quarter 2016 [188]: low transaction costs are a feature of blockchain payments which can be seen when compared with other payment methods in Table 8.1, so this would be beneficial for both donors and the charities.

Table 8.1: Comparison of Transaction Fees

Transaction Method	**Fee BTC**	**Fee USD**	**Fee GBP**	**Speed**
Bitcoin(average 645 bytes)[a]	0.0001	\$0.25	£0.20	roughly 50 minutes [56]
Western Union[b]	-	\$14.83	£8.90	less then 1 hour
Western Union[b]	-	\$11.50	£6.90	next day
MoneyGram[b]	-	\$16.50	£9.90	less then 1 hour
Ria[b]	-	\$10.00	£6.00	same day

[a] Bitcoin transaction fees are calculated on transaction size, not monetary value [53].
[b] Based on remittance transfer of 120 GBP from the United Kingdom to Uganda [189].

3. Donation speed: All Bitcoin transactions are broadcast immediately in the Bitcoin peer-to-peer network. Each transaction that is included in a valid mined block and added to the blockchain is called a confirmation. A single confirmation takes just over seven minutes (also called median confirmation time) [53, 56]. With each subsequent block mined, the number of confirmations for that particular transaction increases by one. Confirmations prevent double spending: the more confirmations there are, the more assurance is given to the transaction. It is common practice to wait until at least six confirmations have been added to the blockchain [147], taking roughly fifty minutes [56]. Depending on the nature of the transaction or the value, it may not be necessary to wait for six confirmations: a transaction could be considered to be complete after only one or two confirmations. This is fast compared to existing money transfer methods which could take several days [202]: see Table 8.1 for examples. These confirmations also provide a mechanism for the donor to be able to verify if and when a payment was received.

4. Donation provisioning: Provisioning the donations to beneficiaries can be a major challenge, especially in difficult environments. For example, humanitarian financial aid distribution in warzones can be severely hampered if the country's banking system is subject to sanctions; the use of Bitcoins would bypass the need to involve the banking system and ensure that donations reach their intended target, without requiring the charity to transport large amounts of cash [67].

To capitalise on these benefits, and harness the desirable security properties of blockchain solutions, we now discuss our proposal for a new blockchain philanthropic model.

8.2.1 Blockchain Properties

In addition to the benefits discussed above, the technology provides the following security features inherent to blockchains.

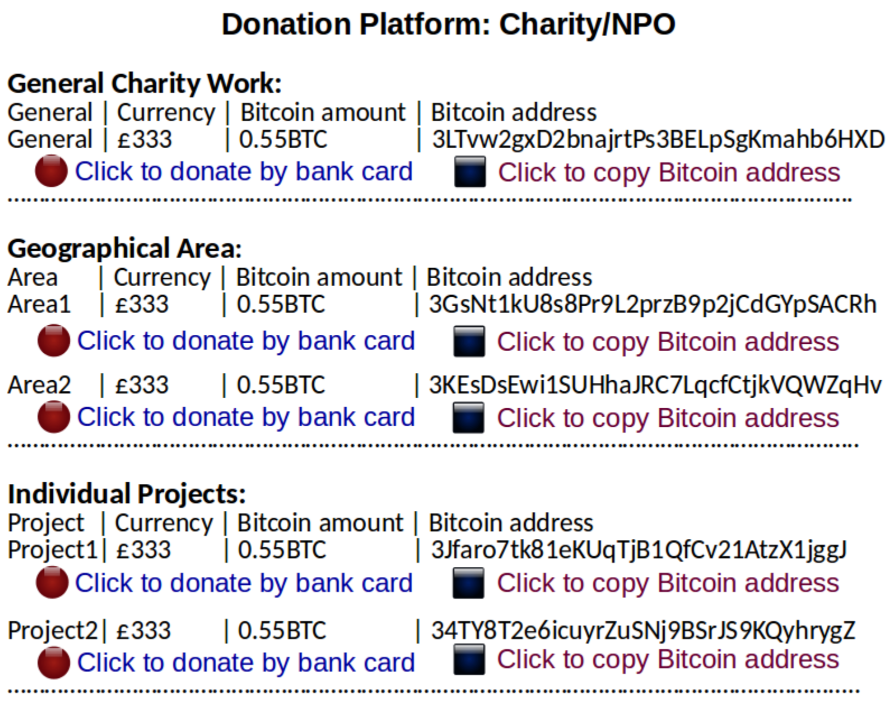
Immutability: Blockchains use cryptographic primitives such as: secure hashing and digital signatures to chronologically record every transaction on the blockchain. This security feature makes it practically impossible for an attacker to manipulate a record that has already been recorded onto the blockchain, leaving an immutable historical record of transactions.
Trust: In a blockchain based architecture, trust assumptions are not placed on one particular entity, instead trust is placed on the underlying cryptography which keeps the blockchain secure. This feature makes a blockchain an ideal platform to interact with entities that do not have established trust relationships.
Reliability: Form a security point of view, a blockchain keeps a shared distributed edger without storing the ledger in a centralised location. This prevents the 'single point of failure' problem as the information recorded in the ledger is distributed throughout the peer-to-peer network. This feature makes a blockchain reliable against denial of service attacks. Furthermore, blockchains leverage consensus mechanisms to establish a majority vote when it comes to updating the ledger with a new block. The shared distributed ledger can be considered as a single record of truth.
Transparency: Blockchains such as the Bitcoin, publish every transaction on a public ledger. This provides a transparent audit trail of transactions, ownership of assets, etc. passing from one entity to another. This feature can be used to validate information recorded on the blockchain, prevent double spending, etc.

8.3 The Blockchain Philanthropic Model

We propose a system where a donor can make their donation in Bitcoin via a Donor Platform, direct to the Bitcoin addresses for individual charity projects. The charity then uses these donated funds to allocate financial aid to individual beneficiaries using hosted Bitcoin wallets that the charity maintains centrally. Beneficiaries can then perform Bitcoin transactions for day-to-day activities. However, as we shall see later in this chapter, not everyone will have the technical infrastructure available to use web-based methods for transactions, and we will also extend the proposal for use in offline geographical areas.

8.3.1 Donor Platform

The web-based donor platform is a public interface between the charity and the blockchain. We consider the charity to have expert understanding of how to efficiently utilise their received donations, so the charity selects causes / projects for donors to choose from. Donors select projects from this list before making a donation in Bitcoin or fiat currency: the charity can choose how 'granular' the web portal list can be: it could range from one Bitcoin address per beneficiary through to a central Bitcoin address for the charity as a whole. Whenever the expense target of a particular beneficiary or a group is reached, the entry is removed and the advertised list is updated. An example list for the donation platform is illustrated in Figure 8.1. Bitcoin donors can obtain the Bitcoin address of the charity/ project then use any Bitcoin wallet/client to donate, or use fiat currency that gets converted to Bitcoin automatically by using an online exchange. Once a donation is made the donor can check the Bitcoin transaction corresponding to the donation on the Bitcoin blockchain. The donor can also query the Bitcoin blockchain to find out whether the donated funds have been used or not.



Donation Platform: Charity/NPO

General Charity Work:

```
General | Currency | Bitcoin amount | Bitcoin address
General | £333     | 0.55BTC        | 3LTvw2gxD2bnajrtPs3BELpSgKmahb6HXD
   ● Click to donate by bank card    ■ Click to copy Bitcoin address
```

Geographical Area:

```
Area     | Currency | Bitcoin amount | Bitcoin address
Area1    | £333     | 0.55BTC        | 3GsNt1kU8s8Pr9L2przB9p2jCdGYpSACRh
   ● Click to donate by bank card    ■ Click to copy Bitcoin address
Area2    | £333     | 0.55BTC        | 3KEsDsEwi1SUHhaJRC7LqcfCtjkVQWZqHv
   ● Click to donate by bank card    ■ Click to copy Bitcoin address
```

Individual Projects:

```
Project  | Currency | Bitcoin amount | Bitcoin address
Project1 | £333     | 0.55BTC        | 3Jfaro7tk81eKUqTjB1QfCv21AtzX1jggJ
   ● Click to donate by bank card    ■ Click to copy Bitcoin address
Project2 | £333     | 0.55BTC        | 34TY8T2e6icuyrZuSNj9BSrJS9KQyhrygZ
   ● Click to donate by bank card    ■ Click to copy Bitcoin address
```

Figure 8.1: Example List for the Donation Platform

We propose the use of two different methods for processing Bitcoin payments for the charity: Multi-Signature Addresses and Smart Contracts. These will now be described.

8.3.2 Bitcoin Transaction Methods

We propose that donations can be used by the charity for donation provisioning and subsequent SMS payment processing via one of two Bitcoin payment methods: Multi-Signature Addresses and Smart Contracts. When using the RSK Smart Contract, the charity must convert Bitcoins to SmartBTC as described below.

8.3.3 Multi-Signature Addresses

In this option, the Bitcoin addresses of the charity / project are constructed using a methods called multi-signature process. Here, in order to authorise a Bitcoin transaction more than one private key is needed. For example, a 2-of-3 multi-signature is when a Bitcoin address is associated with three private keys and at least two out of the three private keys are needed to authorise a Bitcoin transaction. In our proposal, the multi-signatures are processed using 'Pay To Script Hash' (P2SH) transactions.

First, the multi-signature generation involves using a Full Redeem Script which includes details of the three public keys. This is then hashed to generate a hashed Redeem Script which becomes the P2SH multi-signature. The address the donor makes a donation to is this multi-signature address. All the three key holding entities share the Full Redeem Script between them. The Redeem Script can be used to verify the correctness of the transferred amount and whether the transfer has been sent to the correct multi-signature. Furthermore, the Redeem Script can be used to identify how many signatures are needed to make a payment. To spend the received Bitcoins, the recipient has to provide the full Redeem Script. Some services use this technique to enhance security of hosted Bitcoin wallets e.g. Bitgo [26].

8.3.4 Smart Contracts

A Smart Contract can be defined as a set of instructions represented in computer code published on a distributed network, that receives inputs, execute instructions and provide outputs. The multi-signature scheme described above can be classed as a very low level smart contract, but it doesn't have the capability to carry out further instructions in addition to simple Bitcoin payment transactions. A smart contract could give the charity additional functionality to extend its offerings for both the beneficiaries and donors through features such as: issuing donations to the beneficiaries routinely or whenever they are low in cash, issuance of small micro-finance loans to certain beneficiaries / merchants, record keeping of repayments, donation requests to donors and sending automatically generated audit reports of a charity activity to a regular donor is now possible with the smart contract.

For this added functionality, a more advanced smart contract is needed, but as the Bitcoin blockchain was initially invented as a distributed payment platform, running advance smart contracts on the Bitcoin network is not possible. A suitable platform to run smart contract would be Rootstock (RSK) [4]. RSK is a solution that adds functionality to run advanced smart contracts on the Bitcoin network. RSK is a sidechain that is based on a 2-way peg mechanism [4]. The 2-Way peg is a method to convert Bitcoin Currency (BTC) into Smart Bitcoin Currency (SBTC) and vice-versa. When a user intends to convert BTC to SBTC, some BTC are locked in Bitcoin and the same amount of SBTC is unlocked in RSK and vice-versa [4]: this results in a major advantage that it matches Bitcoin to its native currency SBTC in the sidechain, whereas in other sidechains the native currency is not mapped to Bitcoin.

The instructions to be included in the RSK smart contract can be coded using Solidity which is an object-oriented programming language. The smart contract is then published in the RSK network. This means that the contract exists on every node joining the network, including miners. The instructions on the smart contract can be executed by the charity by broadcasting a message to the RSK network. Similar to the Bitcoin network, a small transaction fee is paid to execute the smart contract. The charity can pre-define which party is liable for this transaction fee. To provide additional security and manageability of the donated funds, similar to the multi-signature process in Bitcoin, a smart contract can also be instructed using programmable logic to receive two or more signatures before a transaction can be executed and broadcast to the peer to peer network.

8.4 Constraints in a Blockchain Based Solution

This section discusses potential constraints that need to be considered before using the proposed blockchain philanthropic model in the real world.

Blockchain based schemes have two main constraints, such as: 1) requiring a well established connection to the Internet as blockchain architectures are based on peer-to-peer networking and new transactions are broadcast to the network. 2) requirement for compatible devices such as computers, tablets or smart phones. This is because to make a Bitcoin transaction a number of cryptographic processes needs to be completed. These include: secure hash generation, generating digital signatures and also storing cryptographic keys securely. So devices that are capable of carrying out at least these functions are required.

There are online wallet providers who will keep the most up-to-date blockchain to verify and forward transactions on behalf of registered users, using a multi-signature process for added security: these online wallets are also called Hosted Wallets. However, a user who wants to make a payment needs to be online in order to access their online wallet using a web browser or via a smart phone application.

People in a geographical area, community or even a country where there is not sufficient network infrastructure to provide a reliable Internet facility would find it difficult to use a hosted wallet. A report from the International Telecommunications Union has identified that more than half of the world's population is not using the Internet [111]. Therefore, a solution that uses basic devices and an already existing communication infrastructure (i.e the mobile phone network) is needed. This will have the advantage that it does not require the users to have an Internet connection: there is more GSM network coverage than Internet access in most countries around the world [201, 111], and the use of mobile phones within the GSM network coverage is considerably higher compared to other communication technologies [111]. Our proposal is discussed in the next section.

8.5 The Philanthropic Model in a Disconnected Environment

In this section, we discuss how the blockchain philanthropic model can be applied to the following use case: humanitarian aid in disconnected environments such as warzones, disaster areas, or economically deprived areas of the world. A charity/NPO will face many operational and security problems distributing financial aid in these challenging conditions. In an environment with poor Internet infrastructure, resource-limited devices and an unbanked community, conventional internet-based money transfer will not be possible and physical cash handling is fraught with security issues.

In Section 2.2.3 of the thesis, we identified a few solutions that were attempting to integrate Bitcoin payments to SMS based payment systems. As we can see, none of these existing solutions is suitable for the offline environment under discussion. Therefore, we propose a novel SMS based mobile payment system that gives capability of making Bitcoin payments. The proposed payment system acts as a gateway to transact with the blockchain, using Bitcoin wallets hosted on beneficiaries' behalf by the charity. Offline beneficiaries can then make and receive Bitcoin payments using SMS messaging on basic "feature" phones along with a One Time Password (OTP) security token. The use of the security tokens guarantees that only a authorised user can send an SMS to make a payment, providing some assurance to the genuineness of the transaction.

Any payment system must meet basic security requirements and deal with potential adversaries. Before the proposal is described in depth, these will now be discussed.

8.5.1 Security Requirements of the Proposed System

We identify the below mentioned security requirements to be satisfied by the proposed system.

1. **Confidentiality**: Sensitive information should not be disclosed to unauthorised parties, whether during processing, in transit, or at rest.

2. **Integrity**: Information must not be tampered with by unauthorised parties when it is in transit or at rest; the system must perform its tasks without unauthorised manipulation.

3. **Authentication**: All participants in a transaction must be authorised and all transaction data must be genuine.

4. **Non-repudiation**: None of the participants in a transaction can subsequently deny taking part in it.

5. **Availability**: A service should not be denied to authorised entities: for example, through Distributed Denial of Service (DDoS) attacks.

8.5.2 Adversarial Model

In a humanitarian aid setting, the adversarial model the proposed solution will encounter is as follows [98]:

1. **State Level Attackers (SL):** Adversaries with high level skills/resources employed by government agencies to attack commercial or government systems. State sponsored cyber attacks on humanitarian operations have been recorded. Depending on the political situation in the charity's area of operation, these attackers might aim to destabilise area/ disrupt operations by targeting the donor platform, the LO/HQ/BPS or the SMS system via the MNO/ GSM attacks.

2. **Cyber Criminals (CC):** are organised groups who attack systems for money, who also have high levels of skill and resources. These may wish to target the local SMS system where transactions are initated, or the BPS where they are processed, in order to make fraudulent transactions. Cyber criminals might also target the web server where the donation platform is hosted to change published content such as: inserting Bitcoin addresses belonging to the criminals by replacing the charity's Bitcoin addresses.

3. **Hacktivists (Ha):** have moderate skills and resources and use digital tools to mount attacks for ideological reasons. A hacktivist might not approve of the charity's ethos and objectives, so may aim to vandalise the donation platform, but should not be able to affect the blockchain: they are unlikely to attack local SMS system

4. **Insiders (In):** may have low levels of technical skill and resources, corrupt users, charity workers or merchants can be particularly dangerous if they have privileged access to data.

8.5.3 Proposed SMS based Bitcoin Payment Scheme

We now discuss the proposed SMS based payment scheme. The charity first creates Hosted Wallets for the beneficiaries. During a secure registration process at the local office, the beneficiaries are issued with OTP tokens that will be used to make payment requests. The proposed scheme interacts with a number of entities that we describe below. The relationship between entities is illustrated in Figure 8.2.

- **Bitcoin Payment Server (BPS):** this is part of the Charity's technical infrastructure that manages the hosted Bitcoin Wallets on behalf of beneficiaries. It securely holds Bitcoin keys for each account holder and is connected to the Bitcoin/RSK peer-to-peer network. It also checks and signs payment requests received from the SMS-Gateway, and once these have been authorised by one of the other key holders, the BPS broadcasts them to the Bitcoin peer-to-peer network.

- **Blockchain:** the distributed ledger shared between the nodes connected to the Bitcoin peer-to-peer network.

- **Charity Local Office (LO):** the Charity has a local office in the disconnected environment. The LO registers phone numbers of users during a secure registration process, and manages distribution of OTP security tokens.

- **Charity Head Quarters (HQ):** The Charity HQ may not be in the same geographically area of the disconnected environment. It has connectivity to the Internet and it is a secure server that holds relevant Bitcoin private keys for all payers.

- **Charity Head Quarters Backup Server (HQB):** this is a secure backup server which also holds relevant Bitcoin private keys for all payers.

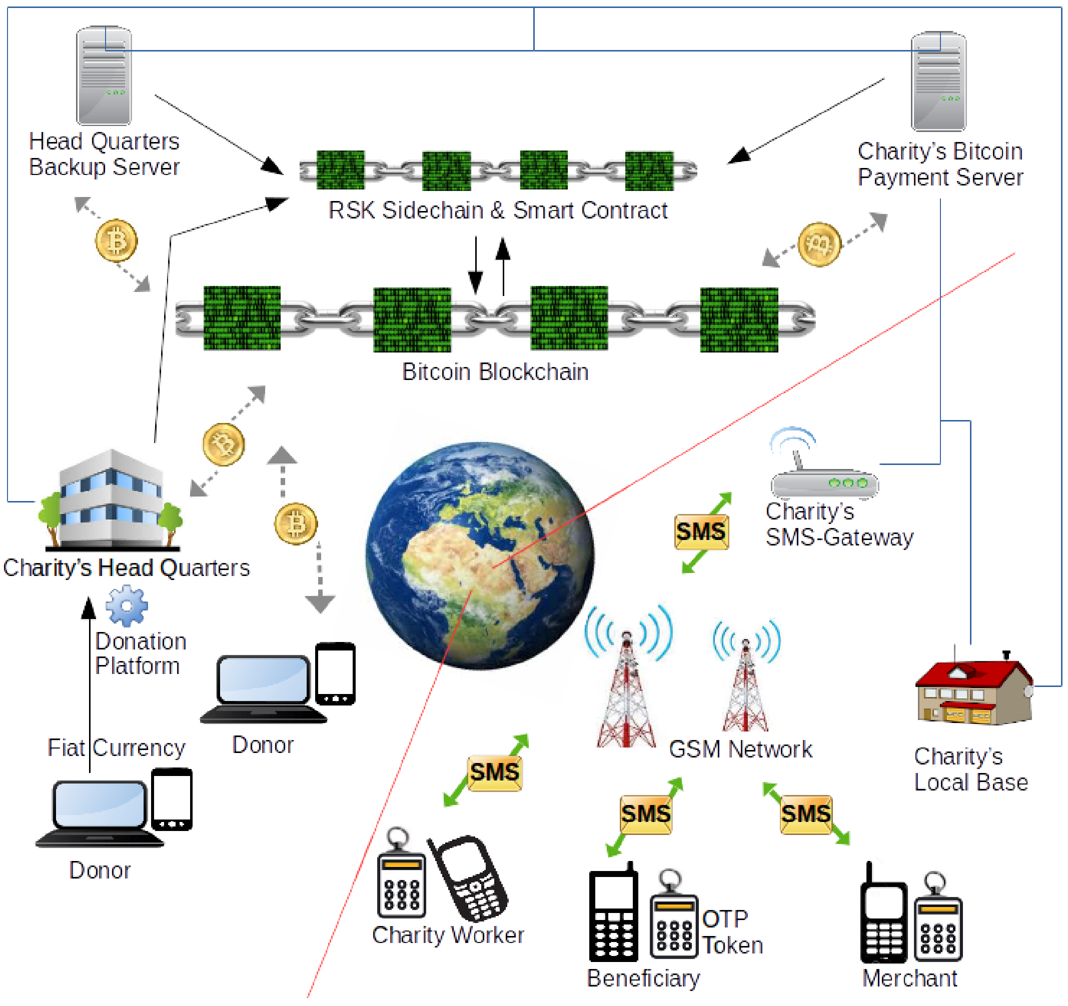


Figure 8.2: Philanthropic Model and SMS Payment System Architecture

- **One Time Password (OTP) Token:** this is a cheap Hash-based One Time Password (HOTP) security token [134] that needs to be used every time an SMS transaction is made. The algorithm that is used to generate the OTP is synchronised between the BPS and each individual security token. Sample OTP generation algorithms can be found in [143, 101].

- **SMS-Gateway:** server that sends and receives SMS transmissions to and from the telecommunication network. All SMS messages used in the proposal are within the standard 160 character length. The SMS-Gateway is connected to the telecommunication network and the BPS, and the BPS is connected to the Bitcoin peer-to-peer network.

- **Donor Platform** Donors select Bitcoin addresses from a web based donor platform, and can use a Bitcoin wallet/client or fiat currency to donate.
- **Payer/ Recipient** Users of the system can make payments (Payer) to any other registered user (Recipient).

Additionally, we make the following assumptions in our proposed scheme:

1. **Charity Head Quarters (HQ):** The charity provides humanitarian aid for beneficiaries in disconnected environments. The charity operates internationally and revenue comes from donations made via a web-based donor platform. It is a reputable and trusted entity, with secure premises and online access/ backup servers which may be geographically distant from the aid environment.
2. **Donors:** Potential donors have capability of accessing the Internet in order to make donations via the donor platform.
3. **Donor Platform:** Hosted on a secure web server adhering to industrial standard security controls to prevent attacks such as: Denial of Service, website defacing, content manipulation, etc.
4. **Bitcoin Payment Server (BPS):** Secure server dedicated to process Bitcoin payments. The server prevents attacks by adhering to industrial standard security controls. Strong encryption is used to securely store all the cryptographic keys to minimise the risk of data breaches.
5. **Phones:** All users of the system possess simple mobile phones ('feature phones'). The knowledge of security code/access PINs to use the mobile phones are only known to their associated owners. The local existing GSM network can be used for SMS messages.
6. **Secure Registration:** During a secure registration process at the LO, the following procedures take place:
    - employees, potential beneficiaries and local merchants will register their mobile numbers and be issued with cheap Hash-based One Time Password (HOTP) security tokens [134] to use with the proposed system. Sample OTP generation algorithms can be found in [143, 101].
    - Demonstrations and required training for using the payment system is provided to the beneficiaries during registration.

- the mobile numbers and OTP security token IDs of the employees and beneficiaries are passed on to the Bitcoin Payment Server (BPS) (via an SMS-Gateway if necessary).
- mobile numbers are assigned a Bitcoin wallet (stored online on the BPS) to be used in subsequent transactions.
- all registration details are forwarded to the BPS (encrypted using the LO's private key), in batches if the LO's internet connection is intermittent

7. **Security Token:** This is a cheap hardware security token that is used every time an SMS transaction is made. When the user requests ("event-driven"), the token device generates a HMAC-Based (HOTP) passcodes. These codes remain valid until used by the authenticating application. Time-Based (TOTP) tokens generate new codes automatically after a set period of time: this approach is not suitable for use with SMS messages that may be subject to potential delays in the messaging system. Standardised HOTP algorithms such as RFC4226 [166] can be used to generate OTPs. The OTPs are normally 8 digits or 6 alphanumeric characters long. The same OTP of a user's transaction can also be generated by the BPS.

8. **Trust:** The SMS-Gateway and BPS are assumed to be trusted & secure. Mobile phones are not.

9. **Bitcoin Wallet Addresses:** All the Bitcoin wallet addresses for users of the system are generated and all the back-end payment transactions based on Bitcoin are made using a 2-of-3 multi-signature transaction process. Three separate keys from three different entities are used to create the Bitcoin address: the key holding entities are the BPS, charity HQ and HQB. This ensures that a payment request cannot be broadcast to the Bitcoin peer-to-peer network by the BPS on its own.

8.5.4 Processing a Bitcoin Payment Request

Payments can be made from charity worker to beneficiary, beneficiary to merchant, or merchant to merchant. Merchants could use an existing Bitcoin address, registered and associated with a short Merchant ID by the BPS, used instead of Ph_P/ Ph_R in transactions. The SMS payment message flow of the scheme is shown in Figure 8.3 and the notation used in the following descriptions is shown in Table 8.2. The security credentials for each entity are shown in Table 8.3 and the content of each SMS messages used is shown in Table 8.4. For simplicity of exposition, the following description shows

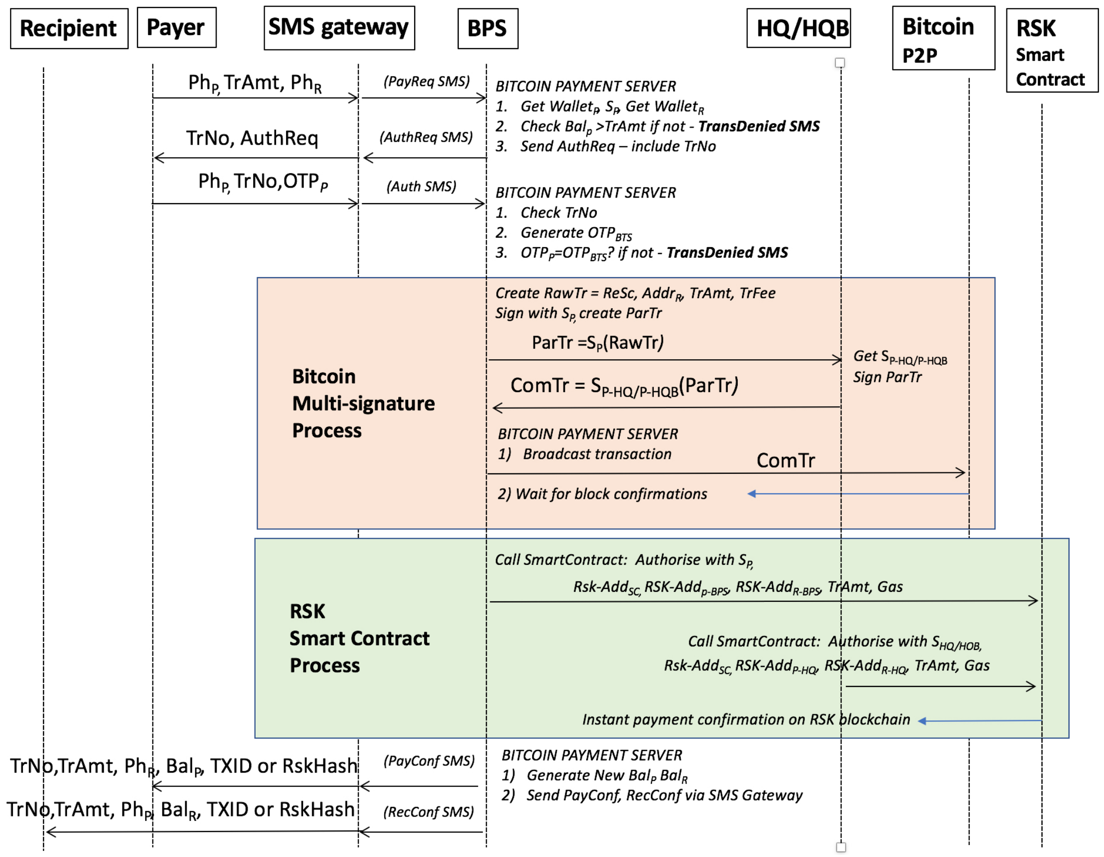

Recipient
Payer
SMS gateway
BPS
HQ/HQB
Bitcoin P2P
RSK Smart Contract
Ph_P, TrAmt, Ph_R
(PayReq SMS)
BITCOIN PAYMENT SERVER
1. Get Wallet_P, S_P, Get Wallet_R
2. Check Bal_p >TrAmt if not - TransDenied SMS
3. Send AuthReq – include TrNo
TrNo, AuthReq
(AuthReq SMS)
Ph_P, TrNo, OTP_P
(Auth SMS)
BITCOIN PAYMENT SERVER
1. Check TrNo
2. Generate OTP_BTS
3. OTP_P=OTP_BTS? if not - TransDenied SMS
Bitcoin Multi-signature Process
Create RawTr = ReSc, Addr_R, TrAmt, TrFee
Sign with S_P, create ParTr
ParTr =S_P(RawTr)
Get S_P-HQ/P-HQB
Sign ParTr
ComTr = S_P-HQ/P-HQB(ParTr)
BITCOIN PAYMENT SERVER
1) Broadcast transaction
ComTr
2) Wait for block confirmations
RSK Smart Contract Process
Call SmartContract: Authorise with S_P,
Rsk-Add_SC, RSK-Add_p-BPS, RSK-Add_R-BPS, TrAmt, Gas
Call SmartContract: Authorise with S_HQ/HOB,
Rsk-Add_SC, RSK-Add_P-HQ, RSK-Add_R-HQ, TrAmt, Gas
Instant payment confirmation on RSK blockchain
TrNo,TrAmt, Ph_R, Bal_P, TXID or RskHash
(PayConf SMS)
BITCOIN PAYMENT SERVER
1) Generate New Bal_P Bal_R
2) Send PayConf, RecConf via SMS Gateway
TrNo,TrAmt, Ph_P, Bal_R, TXID or RskHash
(RecConf SMS)

Figure 8.3: SMS Payment Message Flow

the Head Office (HQ) providing the second Bitcoin key.

Stage 1: Payment Request: The Payer (P) types an SMS which includes payment instructions (*PayReq SMS*) in order to make a payment and sends it to the local phone number provided by the charity. The same phone number is used by all the charity workers, beneficiaries and merchants in the geographical area. When the GSM network receives an SMS payment request, it is forwarded to the charity's BPS via the SMS-Gateway.

Once the message is received, the BPS retrieves Bitcoin wallets for both Payer and Recipient. Following this the BPS checks whether the *TrAmt* is not greater than BAL_P, pseudo-randomly generates a three-digit number, unique *TrNo* and requests for the Payer's OTP by sending *AuthReq SMS*.

To generate the OTP, the Payer presses a button on the OTP token. Following this, the Payer sends the *Auth SMS* containing the newly generated OTP to authorise

Table 8.2: Notation used in Proposed SMS Payment Scheme

Notation	**Description**
$Addr_X$	Bitcoin Multi-signature Address for entity X
BPS / Ph_X	Bitcoin Payment Server(entity) Phone Number of entity X
BAL_X	Bitcoin balance in Account AC_X for entity X
E_K(Z)	Encryption of data Z with key K
X→Y	Message sent from X to Y
HQ / HQB	Head Quarters (entity) / Head Quarters Backup Location (entity)
LO / P / R	Local Office (entity) / Payer(entity) / Recipient(entity)
OTP_X	One Time Password generated by entity X
PK_X/ SK_X	Public/ Secret Key pair of entity X
S_X / *TrHash*	Bitcoin Private Key of entity X (signing key) / Transaction Hash
TrAmt / *TrNo*	Transaction Amount / Transaction Number
TXID	Unique Transaction ID of a transaction recorded in the blockchain. Also referred to as the Transaction Hash (TrHash)
(Z)$Sign_K$	Signature on data Z with signature key K
TrFee	Transaction Fee paid to the Bitcoin miner
RawTr	Raw Transaction created for signing
ParTr	Partial Signed Transaction created after signing *RawTr*
ComTr	Complete Signed Transaction created after signing *ParTr*
ReSc	Full Redeem Script used for the Bitcoin multi-signature address
RSKHash	Rootstock Transaction Hash
$RSK\text{-}Add_{SC}$	RSK Smart Contract Address, unique for the contract and never changes
$RSK\text{-}Add_{X-Y}$	RSK public key (RSK address) of entity X kept securely with entity Y
$S_{RSK-X-Y}$	RSK private key of entity X kept securely with entity Y
Gas	Transaction fee paid to execute instructions on the smart contract

the transaction.

Once the message is received by the BPS, it checks the TrNo to be correct and generates the OTP_{BPS} and compares to the received OTP_P. If the two OTPs match, then the BPS checks that the transaction amount *TrAmt* is not greater than the Payer's balance BAL_P. If any the above mentioned checks fail, *TransDenied SMS* is sent to the Payer declining the transaction. If all checks are passed then the BPS proceeds to making a Bitcoin payment, using one of the two proposed options.

Stage 2: Bitcoin Transaction Processing

- **Option 1: Multi-signature Process:** We now discuss the first method how a

Table 8.3: Credentials Used in Proposed SMS Payment Scheme

Entity	**Keys and Other Assets**
Payer/ Recipient	No keys, PIN for phone, HOTP token (no PIN) for making payments
BPS	S_{P-BPS},$Addr_{P-BPS}$, $Addr_{R-BPS}$, PK_{LO}, Ph_X, OTP_X
HQ	S_{P-HQ}, S_{RSK-HQ}, ReSc
HQB	S_{P-HQB}, $S_{RSK-HQB}$, ReSc
LO	SK_{LO}, Physical OTP tokens, phone numbers (payers/recipients), plus registration details/ OTP allocation details
Donor	S_{Donor}/ V_{Donor}
Donor Platform	$Addr_{Project}$

Table 8.4: SMS Payment Messages

Message	**Content**
PayReq SMS	Ph_P, TrAmt, Ph_R
AuthReq SMS	TrNo, AuthReq
Auth SMS	Ph_P, TrNo, OTP_P
TransDenied SMS	Ph_P, TrNo, Ph_R, Denied
PayConf SMS	TrNo,TrAmt, Ph_R, BAL_P, TXID
RecConf SMS	TrNo,TrAmt, Ph_P, BAL_R, TXID
PayConfRSK SMS	TrNo,TrAmt, Ph_R, BAL_P, RskHash
RecConfRSK SMS	TrNo,TrAmt, Ph_P, BAL_R, RskHash

Bitcoin transaction in our proposed payment scheme can be processed. To process the payment request using multi-signature process, the BPS first constructs a Raw Transaction (*RawTr*). The *RawTr* includes the Full Redeem Script (*ReSc*), the receiver's multi-signature address where the payment is going to, *TrAmt* and *TrFee*. At least two key holding entities needs to sign the *RawTr* before a valid transaction to be broadcast to the peer to peer network can be generated. The *RawTr* is first signed by the BPS using the corresponding Payer's private key S_P to generate the partially signed transaction (*ParTr*). The (*ParTr*) is then forwarded to the to the HQ for signing.

$BPS \rightarrow HQ\text{: } ParTr = (ReSc, Addr_R, TrAmt, TrFee)Sign_{S_P}$

Before authorising the payment request, the *ParTr* is first verified by the HQ to validate whether the payment amount and the number of signatures needed is correct. After these checks are successfully validated by the HQ, it signs the *ParTr* using the private Bitcoin key S_{P-HQ} to generate the Complete Signed

Transaction *ComTr*, which is then sends this back to the BPS.

HQ→BPS: $ComTr = (ParTr)Sign_{S_{P-HQ}}$

Once the message is receive, the BPS broadcast the *ComTr* to the Bitcoin peer to peer network. Once broadcast, in order to trace the particular transaction on the blockchain a unique transaction-id (TXID) can be used. The broadcast transaction is then received by the network of Bitcoin miners and the first miner that published the valid block in the Bitcoin blockchain which also includes the transaction also gets paid the specified transaction fees *TrFee* for the payment. The first block that gets added to the blockchain is the first confirmation of the transaction. Before sending the confirmation SMSs, the BPS waits for the transaction to be confirmed in the agreed number of blocks.

- **Option 2: Smart Contract Process:** The second method to process a Bitcoin transaction in our proposed scheme is by using a smart contract. Here, we extend the capability of the proposed system to show how the Bitcoin multi-signature transaction process can be replaced by a Smart Contract for Bitcoin transaction processing. The BPS calls the Smart Contract and authorises the *TrAmt* and the fee for executing the transaction also called *Gas* is paid by using the $S_{RSK-P-BPS}$.

  BPS→RSK: $RSK\text{-}Add_{SC}$,$RSK\text{-}Add_{P}$,$RSK\text{-}Add_{R}$, *TrAmt*,*Gas*

  After the message being broadcast in the RSK network, the HQ or the HQB calls the smart contract. This acts as the second set of instructions needed by the smart contract to execute the transaction. The paid amount *TrAmt* and the transaction fee *Gas* is authorised by the HQ/HQB by using the $S_{RSK-P-HQ/HQB}$.

  HQ/HQB→RSK: $RSK\text{-}Add_{SC}$,$RSK\text{-}Add_{P-HQ}$,$RSK\text{-}Add_{R}$, *TrAmt*,*Gas*

  The smart contract after successfully receiving instructions from both BPS and HQ/HQB, executes the transaction to transfer the value *TrAmt* to the recipient. As a result, unique transaction details related to the transaction are recorded instantly on the RSK blockchain in the format of a hash (RskHash). Transactions are recorded in the RSK blockchain instantly, therefore, the BPS does not need to wait for a transaction confirmation.

Stage 3: Payment Finalisation: Following successful processing of the Bitcoin transaction using one of the methods we discussed above, the BPS updates the payer and the recipient balances. Then the BPS sends confirmation messages via the SMS-Gateway: *PayConf SMS* or *PayConfRSK SMS* to the Payer and *RecConf SMS* or *RecConfRSK SMS* to the Recipient. Included in the confirmation SMSs are the unique IDs: TXID or RskHash that can be used to trace the transaction on the Bitcoin or the RSK blockchains.

8.6 Analysis

In Section 8.5.3 we discussed how the proposed SMS-based payment scheme can be used by the charity to provision donations and how it can be used by the beneficiaries to carry out friction-less day-to-day payment transactions in a disconnected environment. In this Section, we discuss SMS security and analyse the proposed SMS-based payment scheme against the security requirements identified in Section 8.5.2.

A summary of targets that adversaries may attack along with suggested counter-measures is shown in Table 8.5.

8.6.1 SMS Security Issues

The SMS system has well documented security issues. The SMS service is vulnerable to man-in-the-middle attacks and spoofing because SMS messages are not encrypted by default. Interception/redirection using false base stations in GSM networks, eaves-dropping at the Short Message Service Centre (SMSC), and SS7 hacking [97] are a few attack methods. SMS fuzzing techniques (the “SMS of Death”) can also be used for denial of service and other attacks [145]. Adversaries such as: *SL*, *CC* and *In* might target the SMS system aiming to make fraudulent transactions. Even though these issues are not addressed directly in our proposed scheme, measures have been included which provide some deterrent to would-be attackers.

In the proposed scheme, the use of OTP prevents an attempts of replay attacks. Furthermore, because the charity sends out the *AuthReq SMS* before any payment, alerts the users of any potential fraudulent transaction. The *TXID*/*RSKHash* included in the confirmation SMSs provide further assurance and can be used to cross check with the Bitcoin/RSK blockchain. The inclusion of the *TrNo* on the confirmation messages means that during a face-to-face transaction, Payer and Recipient can verbally compare the value as an extra level of assurance that the transaction is correct. The methods we use in our solution provides a higher level of security compared to other SMS based

Table 8.5: Attack Targets, Adversaries and Countermeasures

Target	**SL**	**CC**	**Ha**	**In**	**Countermeasure**
Donor Platform	y	y	y		Hosted on a secure web server adhering to industrial standard security controls to defend against : DDoS, website defacing, content manipulation
HQ/HQB/BPS (DDoS)	y		y		HQ/HQB has secure premises and backup servers: BPS managed under industrial standard security controls and best practices to prevent attacks.
HQ/HQB/BPS (privilege escalation)	y	y		y	Use of security controls such as: access control, routine web-application vulnerability assessment/patching and storing keys encrypted
SMS (MNO/GSM)	y	y			GSM/SMS security issues partially mitigated by OTP 2FA and TXID/RSKhash on confirmations
SMS spoof	y	y		y	OTP/TXID/RSKhash gives some assurance that payment is genuine
SMS replay	y	y		y	OTP prevents replay attacks
Blockchain/RSK (DDoS)	y		y		DDoS attacks not viable in distributed ledger, and integrity is innate in blockchain solutions

Bitcoin schemes. For an example, the Coinapult SMS scheme requires the user to send an SMS containing a security code sent by the payment service in a previous SMS which provides limited assurance that the transaction is legitimate.

8.6.2 Security Requirements

Confidentiality

- **Security of Bitcoin private keys:** The Bitcoin private key for a corresponding Bitcoin address allows the private key holder to transfer those Bitcoins to any new address. If this private key or the Bitcoin wallet that securely stores this key is lost or not accessible, then the Bitcoin value recorded to that Bitcoin address can be considered to be lost. This is because without the corresponding private key the Bitcoins cannot be transferred. The proposed scheme uses 2-of-3 multi-signature process. This avoids the risk involved in losing a Bitcoin private key by allowing any two out of the three private key holders to generate a combined signature in order to transfer the Bitcoins to another address.

- **Donor anonymity:** Because the proposed scheme accepts donations in Bitcoin, a donor who wishes to remain anonymous can make the donation in Bitcoin. However, this may introduce management issues for the charity. This is because some anonymous donations may need special reporting and investigation due to possible money laundering/fraud regulations or other suspicious financial activity. For example, in the UK, anonymous donations over £25,000 have to be reported as a "serious incident" [185]. This could be resolved by a charity policy stating that donations over a certain amount need personal identification to be registered with the charity.

- **Server attacks (HQ/HQB/BPS):** As identified in our adversarial model in Section 8.5.2, attackers such as *SL* and *CC* will aim at infiltrating security keys, transaction data and identity information. Furthermore, *Ha* may wish to find embarrassing data, manipulate content to cause reputational damage. The countermeasures recommended for overcoming these threats are shown in Table 8.5.

Integrity

- **Blockchain:** All transactions are chronologically recorded in the blockchain using cryptography. If an attacker were to change any record in the blockchain which is a globally shared distributed ledger, the attacker has to change the particular block where the transaction is recorded as well as all the consecutive blocks that are recorded after that. This is considered practically impossible. Therefore, it ensures the integrity by providing an immutable record of past transactions.

- **RSK blockchain:** One of the drawbacks of using new peer-to-peer networks is getting wide spread adoption, which helps to improve robustness of the system. The RSK blockchain, uses the existing miners in the most popular Bitcoin peer-to-peer network mining. This provides reliability and robustness of the mining process. Furthermore, the RSK blockchain is also based on proof-of-work similar to Bitcoin to make the process of mining fair. Also, the distributed ledger uses SHA256 hashing and chain of signatures to prevent double-spending similar to Bitcoin. RSK uses a checkpointing service provided by a federation of well-known and respected Bitcoin community members [4].

- **Server attacks(HQ/HQB/BPS/Donor Platform):** *CC* may target the donation platform with the intention of changing the charity's Bitcoin addresses with addresses belonging to the criminals. The content of the servers may be tampered and vandalised by *Ha*. There might be attempts of tampering content by *SL* to undermine the credibility of the charity. Furthermore, *CC*, *SL* might tamper transaction records at the BPS to make fraudulent transactions. See Table 8.5 for a list of countermeasures.

- **SMS replay attacks:** Potential attackers including: *SL*, *CC* and *In* might attempt to replay SMS messages for fraudulent transactions. Our proposed solution counters replay attempts by requesting the payer to include the OTP in the *Auth SMS* message. Once received the BPS before authorising a payment verifies whether the sent OTP is correct. Any attempts to replay previous OTPs will fail.

- **Unauthorised donation trading:** If the donation made in Bitcoin is provisioned directly to a wallet owned by the beneficiary, there may be the risk of the beneficiary trading the received Bitcoins to gain financial profit, instead of using it for spending the money on buying necessities. The donors would not want this kind of use of their donations and it would also bring bad reputation to the charity. The proposed SMS based payment system provides protection against unauthorised trading of received donations by using not issuing the donations to a Bitcoin address owned by the beneficiary. If the beneficiary wants to buy something or transfer some funds to another beneficiary the SMS payment process must be used and the reconciliation is done by the BPS.

- **Non-Repudiation:** Every transaction is carried out using digital signatures and confirmed transactions are recorded on the blockchain. The blockchain provides an immutable audit-trail for every transaction, thus an entity that has participated in a transaction cannot deny involvement at a later stage.

<u>**Authentication**</u>

- **Authenticating the payment request SMS:** As the payment scheme is based on SMS, additional authentication of the payer and the payment request is required: OTP security tokens are therefore used as a two-factor authentication method. Only the rightful owners are able to operate the tokens as the tokens are pass-code protected. The charity's Bitcoin payment server authenticates the user by verifying the OTP included in the SMS. This means that in the event of a phone being lost or stolen, an attacker will not be able to make a valid transaction without having the security token and knowing its pass-code. Network delays will not cause adverse effects on the authentication process as the OTP is valid until the BPS receives and processes it. This provides some protection against spoofing attacks by adversaries such as *Sl*, *CC* and *In*.

- **Mobile phones:** As per our assumption, the handsets are protected by PIN codes. This act as a barrier to attackers who steal the phone. Even a phone is lost or stolen, to construct a valid transaction request the OTP token and the knowledge of its passcode is needed.

- **Transaction number:** In a point-of-sale transaction, where a beneficiary is purchasing a product from a merchant, both parties can compare the Transaction Number received on confirmation messages before a purchased product is handed out. This provides an additional layer of assurance to the users.

- **Social engineering:** This is a targetted attack that is aimed at obtaining privileged access to data at HQ/BPS. To defend against such attacks staff handling systems or engage in day to day organisational processes are required to go through security awareness training. However, in our proposed solution we use a multi-signatures process for processing transactions and therefore, an insider at the BPS/HQ/HQB is not able to transmit a transaction alone.

Availability

- **Recovering lost Bitcoins:** In our proposed solution a multi-signature transaction processing mechanism is used. In the unfortunate event of any one of the three Bitcoin private key holders losing their key the remaining two parties can recover the Bitcoins by generating a combined signature and transferring those Bitcoins to another address.

- **DDoS attacks:** For attackers such as *SL* and *Ha*, the donor platform, HQ/HQB and BPS are attractive targets for carrying out DDoS attacks. The countermeasures we recommend can be found in Table 8.5. However, any DDoS attacks on the blockchains are not viable because the architecture, proof-of-work and the distributed nature of blockchain provides innate security against such attacks [168, 52, 153, 147].

8.7 Summary

The work carried in this chapter first identified challenges faced by charities/NPOs currently, in particular how strengthening public trust in charities and NPOs, could increase donation revenues and help provide a better service for the targeted beneficiaries. We then identified advantages of blockchain based solutions for charities and discussed how these can be employed, even with their potential constraints.

As our first contribution of this chapter, we proposed a new philanthropic model that addresses the identified challenges while leveraging the Bitcoin blockchain through use of a donation platform. For our second contribution of the chapter we proposed a novel SMS based mobile payment system that can be used by charity workers and beneficiaries in a challenging offline environment. A useful feature is that the SMS based mobile payment system runs on the existing GSM network and does not require an Internet connection.

It also uses a OTP based dual authentication method in order to provide assurance that only a genuine payer can make a payment. The system also provides a payment received and payment confirmation SMS messages for the payer and payee. The payment processing on the proposed scheme can be done by either a 2-of-3 multi-signature transaction process with the Bitcoin network, or a smart contract for advanced functionality utilising the RSK network. The use of multi-signature process addresses the issues such as authorisation before payment and recovery of lost Bitcoins. The proposed SMS-based payment scheme was then evaluated for its security.

Chapter 9

Conclusion and Future Work

Contents

The thesis investigated both the EMV based centralised payment and Bitcoin / blockchain based distributed payment architectures. We were able to identify a number of weakness and potential issues that raise payment security concerns. Addressing these concerns in the previous chapters the thesis proposed a number of solutions that improve the security of both centralised and distributed payment transactions. In this chapter we summarise our main contributions of the thesis and conclude the discussion by suggesting future directions.

9.1 Summary and Conclusion

The main aim of the thesis was to enhance the security of centralised and distributed payment transactions. This included, investigating security aspects of centralised and distributed payments and proposing improvements to address identified security concerns and limitations. To facilitate the main aim of the thesis, three main objectives were identified in Section 1.2. The three main objectives of the thesis are:

Objective-1: Investigate payment transactions in both centralised and distributed payments while showing more emphasis on new/emerging payment technologies.

Objective-2 Identify potential weaknesses and concerns in payment technologies that pose a threat to the security of payment transactions.

Objective-3 Propose improvements that address these identified weaknesses and concerns to enhance the security of payment transactions.

In this thesis, all three objectives were successfully achieved. We discuss below how the objectives of the thesis were achieved in our six main contributions of the thesis.

To achieve **Objective-1**, the thesis explored new and emerging payment technologies used in both centralised and distributed payment systems. The payment technologies we investigated include: EMV Chip&PIN, Contactless Card/Mobile Payments, Tokenisation, Digital Currencies and Blockchain. More specifically, in the first and second contributions, the thesis investigated the EMV Chip&PIN and the EMV OPV process. In the third and fourth contribution, the thesis investigated the EMV Tokenisation used in Chip&PIN, Contactless Card and Mobile Payments. The investigation in to payment technologies in contributions one, two, three and four were carried out under improvements for EMV based centralised payments in the thesis.

The fifth contribution, investigated digital currencies, fair-exchange, anonymous payments, bitcoin and blockchain. The sixth contribution, investigated blockchain technologies, distributed ledger technologies, smart contracts, donation payments, international remittance, mobile and SMS payments. The investigation in to payment technologies in contributions five and six were carried out under improvements for bitcoin/blockchain based distributed payments in the thesis.

The improvements proposed in this thesis are divided into two main categories. The first main category that we explore is the EMV based centralised payments. Under this category, we investigated the architectures of EMV based centralised payments, identified potential weaknesses and proposed a number of improvements. The first payment technology that we investigated was the EMV OPV process used in card based payment transactions.

In our first contribution, **Objective-2** was achieved by identifying a number of potential attack scenarios that pose a threat to the security OPV process. One of the main reasons for these concerns is the indelible trust assumptions placed on a number of entities in the payment architecture. Addressing these security concerns, we then proposed a protocol and improvements that enhance the security of OPV process. Furthermore, three encryption methods that can be used to protect the PIN and provide end-to-end security between the payment card and the issuing bank were presented. We also explained how the OPV process can be linked with the online transaction authorisation process to prevent a type of replay attack that we identify. The proposed improvements achieved **Objective-3** in this contribution. We then subject our proposed protocol to mechanical formal analysis and found no feasible attacks. Finally the proposed improvements were implemented to obtain potential performance penalties that the existing OPV process has to bear if our improvements were applied.

The OPV process that we discussed above is carried out in different instance to the online transaction authorisation. However, in our second contribution, we extended our work to explore a second method that the EMV OPV process can be deployed in. In the second method, the OPV and the online transaction authorisation is carried out in the same set of messages. We called this second method, Unified Authorisation. Achieving **Objective-2**, in our research, we were also able to identify a number of potential attack scenarios in the Unified Authorisation method that pose a threat to the security of OPV process and the overall payment transaction. Addressing the security concerns in the Unified Authorisation, we proposed a protocol that improve the security of OPV and the associated payment transaction. The proposed improvements achieved **Objective-3** in this contribution. We then analysed our protocol and implemented the proposed improvement in Unified Authorisation to identify potential performance penalties.

Our work on OPV has identified a number of potential attack scenarios that could lead to significant financial loss if realised. Moreover, the attacks we described can be considered difficult to detect. This is because the authorisation entity would not be able to identify whether the OPV-based transaction was actually made by the cardholder or the adversary at the compromised intermediary. This would mean that, by the time that the compromise is detected, the adversary might be in a position to cause significant damage in terms of financial fraud. Therefore, we strongly suggest that the proposed improvements must be considered by financial institutions, payment schemes, payment solution providers, etc. when deploying EMV OPV in a geographical region.

Another important aspect of EMV payment architecture is the transaction authorisation. In this thesis, we introduced an inherent weakness associated with EMV transactions and discussed how PAN compromise has led to significant financial losses for financial institutions. We then introduced EMV Tokenisation which is being adopted by the payment industry as a solution to PAN compromise. In our third contribution, **Objective-2** is achieved by identifying a bottleneck in the new tokenisation architecture which prevents the usability of secure tokenised payments in offline environments. We also discussed scenarios where having the capability of carrying out fully offline tokenised payments are beneficial for both merchants and consumers. Achieving **Objective-3** of the thesis, we proposed a contactless mobile payment protocol based on EMV tokenisation that supports fully offline transactions while preserving security guarantees. In addition to this, the protocol provides end-to-end encryption to the tokenised payment transaction communication between the mobile and the payment terminal. Furthermore, we extended our protocol to show how ambient sensing can be used even in an offline environment to detect and prevent token relay attacks. The proposed protocol was then subject to mechanical formal analysis for its security and after a successful run of the analysis no feasible attacks were identified. Finally, a practical implementation of the proposed protocol was carried out to obtain performance measurements. During our implementation, the mobile payment application was provisioned to an embedded hardware secure element of a Nokia 6131 mobile phone.

Even though the primary use of EMV tokenisation is to prevent PAN compromise, it also provides a number of other benefits such as: being able to use the token cryptogram for managing financial risks, confining a payment to a single payment channel, appointing a TSP to authorise payments on behalf of the issuing bank, using features such as ambient sensing to prevent relay attacks, etc. We believe that by using our protocol to provide offline capability to secure tokenised payments, this payment technology can expand to untapped markets and improve wider adoption in the payments industry. Besides this, to our knowledge the work carried out in this thesis is the first to propose an offline contactless payment protocol based on tokenisation with mechanical formal analysis and practical implementation, at the time of writing.

In our fourth contribution, we then focused our attention to the security aspects of tokenised payments. After investigating the current tokenisation payment architecture, we were able to identify five main attack scenarios that pose a threat to the security of tokenised payments especially when a static-token is used in every transaction. By identifying these attacks, we were able to achieve **Objective-2** of the thesis under this contribution. Addressing these security issues we propose a protocol that uses Dynamic Transaction Tokens (DTT) to improve the security of payment transactions that are based on EMV tokenisation. More significantly, the protocol provides mutual authentication between the mobile and the terminal, prevents the attacks by using DTT that is unique to a particular transaction and provides end-to-end encryption between the terminal and the TSP as well as the terminal and the mobile. The protocol was then subject to mechanical formal analysis for its security and protocol weaknesses or attacks were identified.

In our proposed DTT based solution, the authorisation entity can gain valuable insights of a potential transaction even before a transaction request arrives at the TSP. This is because a DTT is issued upon request of a particular transaction and due to this the TSP can obtain accurate risk assurance levels for a transaction and detect any fraudulent activity. Therefore, we suggest that our identified attack scenarios are considered by payment solution provides, financial institutions, payment networks and other relevant parties when rolling out static-token based solutions. By proposing an improved protocol, we were able to achieve **Objective-3** of the thesis under this contribution. As an alternative, we suggest our proposed solution based on DTT to be considered to provide improved security assurance to tokenised payment transactions.

The second main category that we explored in this thesis is the Bitcoin/blockchain based distributed payments. In our fifth contribution, achieving **Objective-2**, we investigated the architectures of distributed payment systems and identified potential drawbacks in distributed payment transactions. In our research one of the issues we identified is the problem of establishing fairness in anonymous payment transactions in e-commerce environment. More significantly, Bitcoin is a cryptocurrency that is widely used to provide anonymous payments and has the largest market capitalisation at the time of writing. Because of the anonymous nature of Bitcoin payments, however, guaranteeing fair-exchange during e-commerce transactions is a major problem.

The blockchain technology is expected to be a disruptive technology in the payments industry by opening new pathways to payment innovation. In such times where the market is looking forward for a wider adoption of payment solutions based on blockchain technology, issues such as not being able to establish fair-exchange in e-commerce can be considered a deterrent. Therefore, addressing the aforementioned issue, we proposed an anonymous fair-exchange payment protocol that establishes strong-fairness in an e-commerce transaction when anonymous Bitcoin is being used as a payment method. The involvement of a TTP in certain scenarios can be considered a bottleneck. In our proposed protocol we keep the involvement of a TTP to a minimum by using a offline TTP that only intervenes in the protocol if a transacting party misbehaves, prematurely aborts the protocol or a communication failure happens.

Furthermore, we then outlined a few issues related to the transaction link-ability in

Bitcoin and showed how the protocol can be extended to support Zerocoin/zerocash to provide improved anonymity and security guarantees. We explained how the protocol can be used with other cryptocurrencies based on blockchains. By using our protocol to facilitate in an e-commerce transaction which involves anonymous payments, transacting parties (both the merchant and the consumer) can engage in the payment transaction with confidence as it provides a guaranteed fair-exchange and dispute resolution within the protocol run. The proposed improvements achieved **Objective-3** in this contribution. We strongly believe that using methods such as our protocol can remove reluctance/uncertainty of using cryptocurrencies as a method of payment in e-commerce transactions and help achieve wide adoption of blockchain based payments.

The use of blockchain technology is not limited to e-commerce payment transactions. The technology can be used in other payment related scenarios. In our research we identified that the philanthropic sector as one of the potential industries to benefit immensely by leveraging the blockchain technology. We then investigated the current philanthropic sector and identified that the public trust on charities are declining and negative media coverage such as [186] has impacted public confidence in charities. One of the main reasons for this is lack of accountability and transparency on how charities collect and spend donations[157].

Following this, in our last contribution, we carried out research to identify how blockchain technology can help charities address these issues and improve donors' trust in the charity sector. We then proposed a novel philanthropic model based on blockchain technology which can be used by a charity to address challenges such as: donation transparency, reduce transactional cost, donation speed, donation provisioning, etc. The donation platform on the philanthropic model lets a donor make a donation in Bitcoin or fiat currency to a selected charity activity. The donor has transparency to his/her donation with the use of Bitcoin public ledger. Furthermore, distributed payment systems such as Bitcoin does not rely on third-party payment clearing services, it can support lower transactional fees especially for cross-border payments. Therefore, the philanthropic model significantly reduces the unnecessary transaction cost of donations. By identifying these weaknesses and concerns, we were able to achieve **Objective-2** of the thesis under this contribution.

We then discuss how our novel philanthropic model can be applied for humanitarian aid in a disconnected environment. We identified that poor Internet infrastructure and resource-limited devices as main constraints in using a blockchain based solutions in such an environment. Addressing these issues, we propose an SMS-based mobile payment system that uses the existing GSM network. We use OTPs generated by OTP token devices to authenticate SMS-based transaction requests. The proposed payment system acts as a gateway to transact with the Bitcoin blockchain. The proposed system provides the capability for the charity to provision the received donations to the beneficiaries. Furthermore, the payment systems also allows the beneficiaries to make secure Bitcoin payments for their day to day transactions by using feature phones, issued OTP tokens and simple SMSs.

In our solution we describe two methods how Bitcoin transactions can be processed in the SMS-based payment system. Firstly, we show how Bitcoin multi-signature process can be used to process transactions. The 2-of-3 multi-signatures provides assurance that a single key holder alone cannot generate a valid transaction. This provides a level of authority as to how the charity could manage payment authorisations. Furthermore, in the unfortunate event of a party losing their Bitcoin private key, the remaining two parties are able to recover the Bitcoins. The second method we use to process Bitcoin transactions is by using a smart contract. In our approach, we show how the same multi-signature process can be applied in a smart contract. As the Bitcoin network is initially invented as a distributed payment platform, it does not support smart contracts. Therefore, we use the Rootstock network and show how Bitcoin payment transactions originating from our SMS-based payment system can be processed. In both processing methods, the payer and the payee receive payment confirmation SMSs from the charity. The proposed solution is then analysed for its security requirements. By proposing a novel philanthropic model and a SMS-based mobile payment system, we were able to achieve **Objective-3** of the thesis.

Another aspect that needs to be considered when it comes to cryptocurrencies based solutions is the price volatility. It must be noted that the Bitcoin exchange value has shown dramatic volatility since early 2013 where the Bitcoin market price has ranged from $12 USD to an all time-high of $19,900 in December 2017. It can be considered that the volatility of Bitcoin exchange value poses a financial risk for the charity. However, using Bitcoin might be the only viable option in an environment where the banking system/economy may have collapsed. The proposed payment system is aimed at a closed eco-system where payments are made within a constrained geographical environment. This to some extent minimises the effects of Bitcoin price volatility. As a long-term solution to this problem, however, the charity can use a private blockchain solution that replaces the Bitcoin blockchain to give more control over exchange rates.

In our conclusion, we discussed how the three main objectives of the thesis were successfully achieved in each of our six main contribution of the thesis. A summary of the key findings of each contribution is listed in Table 9.1.

Table 9.1: Summary Table of Contributions

Contribution. Payment Technology	**Identified Security Concerns, Attacks and Limitations**	**Proposed Solutions and Improvements**
1. EMV OPV	* Two potential attack scenarios in segmented authorisation: correct PIN in OPV message and OPV response message. * Indelible trust assumptions placed on intermediaries.	* Protocol and improvements that enhance the security of OPV by addressing the identified attacks. * Provided end-to-end encryption, replay prevention and other security guarantees.
2. EMV OPV	* Two potential attack scenarios that compromise the security of OPV in unified authorisation: correct PIN in OPV message and PIN block replay in OPV request.	* Protocol and improvements that enhance the security of OPV by addressing the identified attacks. * End-to-end encryption, binding of OPV and transaction authorisation for replay prevention
3. Tokenisation	* limitations in making secure tokenised payments in offline environments.	* Protocol which provides capability of making secure tokenised payments in offline environments.
4. Tokenisation	* Five potential attacks: over charging, capturing static token, capturing unpredictable number, adversary replays an authorisation and replaying an authorisation response.	* Contactless mobile payment protocol that uses DTT to provides: security against the five attacks. * Provided end-to-end encryption, mutual authentication and other security guarantees.
5. Blockchain	* Providing fair-exchange in e-commerce when anonymous payment methods are used. * Concerns related to Bitcoin transaction linkability.	* Protocol that guarantees true fair-exchange when Bitcoin is used as an anonymous payment method. * Extended the protocol to provide improved security and anonymity.
6. Blockchain	* Issues in the current philanthropic model: donation transparency, cost, speed of getting the donations to beneficiaries, provisioning received donations to beneficiaries in disconnected environments.	* Novel philanthropic model that address these challenges by using blockchain technology. * SMS based mobile payment systems to provision donations and to be used by the beneficiaries for day to day payments.

9.2 Suggestions for Future Research

In this section, the thesis makes suggestions for future work and research directions.

The potential suggestions are categorised under each contribution area. The suggestions for future work and research directions related to enhancing the security of OPV are as follows.

It would be beneficial to understand and evaluate the security of the current key sharing schemes between the payment terminal providers and their payment terminals (at merchants' premisses). In the evaluation, aspects such as: trust assumptions, liability guarantees, security solutions used and compliance with payment card industry standards should be taken in to account.

Furthermore, evaluating the payment architecture for contactless payment systems with respect to issues similar to those presented in this study would be another area for future work.

Our next recommendation for future research directions would be the investigation of the proposed next generation EMV specification. The proposed new changes would mean that there will be significant architectural and security framework changes to how EMV based centralised payment are carried out. Understanding how the changes would impact key entities in the payment industry would be beneficial.

The suggestions for future work and research directions related to enhancing the security of tokenised payments are as follows. We would like to explore and evaluating the security of other transaction scenarios related to tokenised based payments. One such example would be: making an offline token payment while only the terminal is online-capable. We would like to investigate how our protocol can be extended to support such payment scenario. Another area is to include additional transaction modes and expand our threat model to include the mobile being compromised by an adversary.

Furthermore, we would like to implement the proposed fully offline protocol proposed on a 32bit secure element on a modern smart phone to compare performance variations. Another point to note is that, most modern smart phones with embedded secure elements have access to their secure elements restricted by manufacturers and finding a device that provides access to the secure element would be challenging. The last research direction we identify is exploring how similar protocols can be designed to work with Host Card Emulation.

The suggestions for future work and research directions related to enhancing the security blockchain based payments and transactions are as follows. More significantly, in relation to anonymous fair-exchange, we would like to make improvements to our protocol in order to support exchange of physical products. In such research attempts, a key challenge will be how to deliver the physical products to the recipient's address without compromising the user anonymity. While using centralised payment methods, few e-commerce retailers including Amazon, provide public delivery cabinet services to collect delivery items. Here, the recipients collect their delivered items using PIN codes sent by the retailer.

As we have already explained in the thesis, providing strong fair-exchange without any arbitration is considered extremely difficult or sometimes impossible. Therefore, another viable area of research would be to investigate the possibility of further reducing the involvement of a significant TTP by using distributed TTPs. Moreover, this aligns with the principles of distributed networks where no trust assumptions are placed on a particular party, rather the trust is placed on the security and robustness of the distributed network/blockchain.

Our next future directions are in relation to the use of blockchains in the philanthropic industry. We would like to investigate the possibility of applying our blockchain-based philanthropic model in an ad-hoc network that replaces the existing GSM network. This would further extend the benefits of using the payment system proposed as part of the philanthropic model in an environment without GSM network coverage or where the local communication system has collapsed.

Bibliography

[1] Bitpesa v Safaricom. [Online]. Available: https://www.bitpesa.co/blog/bitpesa-v-safaricom/. Last visited on 08/04/2018. 37

[2] Coinapult SMS. [Online]. Available: https://coinapult.com/sms/info. Last visited on 08/04/2018. 37

[3] NFC technical specifications, NFC forum. [Online]. Available: http://members.nfc-forum.org/members/specifications/technical_specs/. Last visited on 08/04/2018. 89

[4] Rootstock Platform. [Online]. Available: http://www.rsk.co/. Last visited on 08/04/2018. 151, 157, 170

[5] Scyther script for the dynamic transaction token protocol. [Online]. Available: https://www.dropbox.com/s/euqnwf0ds17zd6v/DTT%20Protocol%20Scyther%20Script.spdl?dl=0. Last visited on 08/04/2018. 122

[6] Scyther script for the extended fair-exchange protocol. [Online]. Available: https://www.dropbox.com/s/4nf4nzdyit2h8le/Extended%20Fair-exchange%20Protocol.spdl?dl=0. Last visited on 08/04/2018. 148

[7] Scyther script for the main fair-exchange protocol. [Online]. Available: https://www.dropbox.com/s/5w2nxa5fwnvuytv/Main%20Fair-exchange%20Protocol.spdl?dl=0. Last visited on 08/04/2018. 148

[8] Scyther script for the offline environment protocol. [Online]. Available: https://www.dropbox.com/s/9pppyx091si3uxk/OTT%20Scyther%20Script%20Final.spdl?dl=0. Last visited on 08/04/2018. 102

[9] International Organization for Standardization, ISO 9564-1:2002, Banking – Personal Identification Number (PIN) management and security – Part 1: Basic principles and requirements for online PIN handling in ATM and POS systems , April 2002. 58, 65, 76

[10] Application Programming Interface, Java Card Platform, October 2003. 102, 103

[11] EMV mobile contactless payment: technical issues and position paper, Version 1.0, EMVCo, LLC, October 2007. 89

[12] Java Card Platform Specification: Classic Edition; Application Programming Interface, Runtime Environment Specification, Virtual Machine Specification, Connected Edition; Runtime Environment Specification, Java Servlet Specification, Application Programming Interface, Virtual Machine Specification, Sample Structure of Application Modules. [Online]. Available: `http://java.sun.com/javacard/3.0.1/specs.jsp`, May 2009. Last visited on 08/04/2018. 68

[13] EMV integrated circuit card specifications for payment systems, Book 1: application independent ICC to terminal interface requirements, Version 4.3, EMVCo, LLC, November 2011. 25, 26, 29, 47, 50, 65, 76

[14] EMV integrated circuit card specifications for payment systems, Book 2: security and key management, Version 4.3, EMVCo, LLC, November 2011. 25, 26, 29, 50, 56, 58, 60, 64, 65, 76, 103, 111, 112

[15] EMV integrated circuit card specifications for payment systems, Book 3: application specification, version 4.3, EMVCo, LLC, November 2011. 25, 26, 29, 50, 64, 65, 76, 110

[16] EMV integrated circuit card specifications for payment systems, Book 4: cardholder, attendant, and acquirer interface requirements, Version 4.3, EMVCo, LLC, November 2011. 25, 26, 27, 29, 50, 55, 65, 76

[17] International Organization for Standardization, ISO 9564-1:2011, Financial services – Personal Identification Number (PIN) management and security – Part 1: Basic principles and requirements for PINs in card-based systems , February 2011. 58, 65, 76

[18] GlobalPlatform system: messaging specification for management of mobile-NFC services, V 1.1.2, GlobalPlatform., December 2013. 93

[19] EMV contactless mobile payment: application activation user interface, Version 1.0, EMVCo, LLC, December 2014. 89

[20] EMV payment tokenisation specification: technical framework, Version 1.0 and Version 2.0, EMVCo, LLC, March 2014. 25, 30, 31, 89, 95, 98, 100, 106, 110, 112, 118

[21] EMVCo mobile contactless: EMV profiles of GlobalPlatform UICC configuration, Version 1.0, EMVCo, LLC, December 2014. 89

[22] Apple Pay. [Online]. Available: http://www.apple.com/uk/apple-pay/, July 2015. Last visited on 08/04/2018. 30, 106, 110

[23] Bitcoin remittance startup 37Coins announces closure. [Online]. Available: http://www.coindesk.com/bitcoin-remittance-startup-37coins-announces-closure/, August 2015. Last visited on 08/04/2018. 37

[24] EMV contactless specifications for payment systems, EMVCo, LLC, March 2015. 25, 29, 89, 97, 116

[25] Android Pay. [Online]. Available: https://developers.google.com/android-pay/, June 2016. Last visited on 08/04/2018. 30, 106

[26] Bitgo. [Online]. Available: https://www.bitgo.com/, 2017. Last visited on 08/04/2018. 156

[27] Bitwala. [Online]. Available: https://www.bitwala.io/send-bitcoin-to-mobile-money-for-free-africa, March 2017. Last visited on 08/04/2018. 37

[28] BTC For SMS. [Online]. Available: http://www.btcforsms.com/, 2017. Last visited on 08/04/2018. 37

[29] GiveTrack: Donation Tracking. [Online]. Available: https://bitgivefoundation.org/bitcoin-charity-2-0-initiative/, March 2017. Last visited on 08/04/2018. 151, 152

[30] Massive Equifax data breach hits 143 million. [Online]. Available: http://www.bbc.co.uk/news/business-41192163, September 2017. Last visited on 08/04/2018. 29, 30

[31] Monzo: Banking application. [Online]. Available: https://monzo.com, 2017. Last visited on 08/04/2018. 110

[32] Revolut: Banking application. [Online]. Available: https://www.revolut.com/, 2017. Last visited on 08/04/2018. 110

[33] Worldwide retail ecommerce sales: The eMarketer forecast for 2016. [Online]. Available: https://www.emarketer.com/Article/Worldwide-Retail-Ecommerce-Sales-Will-Reach-1915-Trillion-This-Year/1014369, June 2017. Last visited on 08/04/2018. 125

[34] R. N. Akram, K. Markantonakis, and K. Mayes. A privacy preserving application acquisition protocol. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pages 383–392. IEEE, 2012. 103

[35] R. N. Akram, K. Markantonakis, and K. Mayes. Pseudorandom number generation in smart cards: An implementation, performance and randomness analysis. In Antonio Mana and M. Klonowski, editors, *5th International Conference on New Technologies, Mobility and Security (NTMS)*, Istanbul, Turkey, May 2012. IEEE Computer Society. 68

[36] R. N. Akram, K. Markantonakis, and K. Mayes. A secure and trusted channel protocol for the user centric smart card ownership model. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on*, pages 336–345. IEEE, 2013. 103

[37] R. Anderson and S. J. Murdoch. EMV: why payment systems fail. *Communications of the ACM*, 57(6):24–28, 2014. 28, 29, 47

[38] Android Develop API Guides. Host-based Card Emulation. [Online]. Available: `https://developer.android.com/guide/topics/connectivity/nfc/hce.html`, 2017. Last visited on 08/04/2018. 92

[39] I. G. Askoxylakis, M. Pramateftakis, D. D. Kastanis, and A. P. Traganitis. Integration of a secure mobile payment system in a gsm/umts sim smart card. *System*, 12:13, 2007. 30

[40] N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, CCS '97, pages 7–17, New York, NY, USA, 1997. ACM. 39, 125

[41] N. Asokan, V. Shoup, and M. Waidner. Asynchronous protocols for optimistic fair exchange. In *Security and Privacy, 1998. Proceedings. 1998 IEEE Symposium on*, pages 86–99, 1998. 40

[42] A. Back. Hashcash - a DoS counter-measure. Technical report, 2002. 36

[43] A. Bahreman and D. Tygar. Certified electronic mail. In *Network and Distributed Systems Security Conference*, pages 3–19, February 1994. 39

[44] S. Barber, X. Boyen, E. Shi, and E. Uzun. Bitter to better how to make Bitcoin a better currency. In A. Keromytis, editor, *Financial Cryptography and Data*

Security, volume 7397 of *Lecture Notes in Computer Science*, pages 399–414. Springer Berlin Heidelberg, 2012. 37

[45] BBC News. US and UK accused of hacking SIM card firm to steal codes. [Online]. Available: http://www.bbc.co.uk/news/technology-31545050, 20th February 2015. Last visited on 08/04/2018. 51, 53, 73, 107

[46] M. Bellare and P. Rogaway. The exact security of digital signatures-How to sign with RSA and Rabin. In *Advances in CryptologyEurocrypt'96*, pages 399–416. Springer, 1996. 102

[47] M. Ben-Or, O. Goldreich, S. Micali, and R. Rivest. A fair protocol for signing contracts. *IEEE Transactions on Information Theory*, 36(1):40–46, 1990. 39

[48] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from Bitcoin. In *Security and Privacy (SP), 2014 IEEE Symposium on. IEEE*, 2014. 141, 143, 145, 146

[49] O. Berkman and O. Ostrovsky. The unbearable lightness of PIN cracking. In S. Dietrich and R. Dhamija, editors, *Financial Cryptography and Data Security*, volume 4886 of *Lecture Notes in Computer Science*, pages 224–238. Springer Berlin Heidelberg, 2007. 26, 50

[50] P. Biddle, P. England, M. Peinado, B. Willman, et al. The darknet and the future of content distribution. In *ACM Workshop on Digital Rights Management*, volume 6, page 54, 2002. 33

[51] S. Birkwood. Is Bitcoin the ideal charity currency or a cause for concern? [Online]. Available: http://www.thirdsector.co.uk/analysis-bitcoin-ideal-charity-currency-cause-concern/fundraising/article/1326549, January 2015. Last visited on 08/04/2018. 151, 152

[52] Bitcoin.org. Bitcoin. [Online]. Available: http://bitcoin.org/en/. Last visited on 08/04/2018. 33, 35, 172

[53] Bitcoin.org. Bitcoin wiki. [Online]. Available: https://en.bitcoin.it/wiki/Main_Page, 2014. Last visited on 08/04/2018. 35, 36, 153

[54] J. Black. Developments in data security breach liability. *The Business Lawyer*, 69(1):199–207, 2013. 53, 73

[55] Blockchain.info. Bitcoin in circulation. [Online]. Available: https://blockchain.info/charts/total-bitcoins. Last visited on 08/04/2018. 35

[56] Blockchain.info. Average transaction confirmation time. [Online]. Available: https://blockchain.info/charts/median-confirmation-time, 2016. Last visited on 08/04/2018. 33, 153

[57] Blockchain.info. Bitcoin market capitalization. [Online]. Available: https://blockchain.info/charts/market-cap, 2017. Last visited on 08/04/2018. 127

[58] M. Blum. How to exchange (secret) keys. *ACM Trans. Comput. Syst.*, 1(2):175–193, May 1983. 39

[59] M. Bond, O. Choudary, S. J. Murdoch, S. Skorobogatov, and R. Anderson. Chip and Skim: cloning EMV cards with the pre-play attack. In *2014 IEEE Symposium on Security and Privacy*, pages 49–64. IEEE, 2014. 25, 111

[60] M. Bond, O. Choudary, S. J. Murdoch, S. Skorobogatov, and R. Anderson. Be prepared: the EMV pre-play attack. *IEEE Security & Privacy*, 2015. 25, 107, 111

[61] F. Brezo and P. G. Bringas. Issues and risks associated with cryptocurrencies such as Bitcoin. 2012. 127

[62] H. Burk and A. Pfitzmann. Digital payment systems enabling security and unobservability. *Comput. Secur.*, 8(5):399–416, Aug. 1989. 38, 40

[63] J. Camenisch, J.-M. Piveteau, and M. Stadler. An efficient fair payment system. In *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, CCS '96, pages 88–94, New York, NY, USA, 1996. ACM. 38

[64] J. Casswell. Displaced populations, humanitarian cash transfers and mobile money. [Online]. Available: http://www.gsma.com/mobilefordevelopment/programme/mobile-money/displaced-populations-humanitarian-cash-transfers-and-mobile-money, February 2017. Last visited on 08/04/2018. 151

[65] M. Centenaro, R. Focardi, F. L. Luccio, and G. Steel. Type-based analysis of PIN processing APIs. In *Computer Security–ESORICS 2009*, pages 53–68. Springer, 2009. 26, 50

[66] Charities Aid Foundation (CAF). CAF WORLD GIVING INDEX 2015, A global view of giving trends. [Online]. Available: https://www.cafonline.org/about-

us/publications/2015-publications/world-giving-index-2015, 2015. Last visited on 08/04/2018. 150

[67] Charities Aid Foundation (CAF). Giving unchained:philanthropy and the blockchain. [Online]. Available: https://www.cafonline.org/docs/default-source/about-us-publications/givingunchained-philanthropy-and-the-blockchain.pdf?sfvrsn=4, 2016. Last visited on 08/04/2018. 33, 150, 152, 153

[68] D. Chaum. Security without identification: Transaction systems to make big brother obsolete. *Commun. ACM*, 28(10):1030–1044, Oct. 1985. 38

[69] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Proceedings on Advances in Cryptology*, CRYPTO '88, pages 319–327, New York, NY, USA, 1990. Springer-Verlag New York, Inc. 36, 38

[70] T. Chothia, F. Garcia, J. de Ruiter, J. van den Breekel, and M. Thompson. Relay cost bounding for contactless EMV payments. In *Financial Cryptography and Data Security*, volume 8975, pages 189–206. LNCS, Springer Berlin Heidelberg, 2015. 25, 100, 111

[71] D. L. K. Chuen. *Handbook of digital currency: Bitcoin, innovation, financial instruments, and big data.* Academic Press, 2015. 33

[72] T. Coffey and P. Saidha. Non-repudiation with mandatory proof of receipt. *SIGCOMM Comput. Commun. Rev.*, 26(1):6–17, Jan. 1996. 39

[73] CoinMarketCap. Cryptocurrency market capitalizations. [Online]. Available: https://coinmarketcap.com/, 2017. Last visited on 08/04/2018. 33

[74] Computerworld.com. Vulnerabilities found in three popular payment terminal models can result in credit card data theft. [Online]. Available: http://www.computerworld.com/article/2504956/security0/payment-terminal-flaws-shown-at-black-hat.html, 26th July 2012. Last visited on 08/04/2018. 107

[75] C. Cordery. Regulating small and medium charities: Does it improve transparency and accountability? *Voluntas: International Journal of Voluntary and Nonprofit Organizations*, 24(3):831–851, 2013. 150

[76] C. J. Cordery, D. Sim, and T. Zijl. Differentiated regulation: the case of charities. *Accounting & Finance*, 57(1):131–164, 2017. 150

[77] F. Corella and K. Lewison. Interpreting the EMV Tokenisation Specification. 2014. 104

[78] P. S. S. Council. Tokenization product security guidelines: irreversible and reversible tokens, Version 1.0, PCI data security standard (PCI DSS), April 2015. 30

[79] B. Cox, J. D. Tygar, and M. Sirbu. NetBill security and transaction protocol. In *Proceedings of the 1st Conference on USENIX Workshop on Electronic Commerce - Volume 1*, WOEC'95, pages 6–6, Berkeley, CA, USA, 1995. USENIX Association. 40

[80] C. Cremers. The Scyther tool: verification, falsification, and analysis of security protocols. In *Computer Aided Verification*. Springer Berlin Heidelberg, 2008. 41, 42, 43, 101, 121, 122, 148

[81] C. Cremers and S. Mauw. Operational semantics of security protocols. In *Scenarios: models, transformations and tools*. LNCS, Springer Berlin Heidelberg, 2005. 41, 42, 43, 101, 122, 148

[82] C. J. Cremers, P. Lafourcade, and P. Nadeau. Comparing state spaces in automatic security protocol analysis. In *Formal to Practical Security*, pages 70–94. Springer, 2009. 40, 41, 42

[83] M. Crowe, S. Pandy, D. Lott, and S. Mott. Is payment tokenization ready for primetime? perspectives from industry stakeholders on the tokenization landscape. Technical report, Federal Reserve Bank of Boston & Federal Reserve Bank of Atlanta, 2015. 30, 31, 89, 106

[84] M. J. Culnan and R. J. Bies. Consumer privacy: Balancing economic and justice considerations. *Journal of Social Issues*, 59(2):323–342, 2003. 125, 126

[85] N. Dalal, J. Shah, K. Hisaria, and D. Jinwala. A comparative analysis of tools for verification of security protocols. *International Journal of Communications, Network and System Sciences*, 3(10):779, 2010. 40, 41, 42

[86] R. Davies. *Public Good by Private Means: How philanthropy shapes Britain*. Alliance Publishing Trust, 2016. 150, 152

[87] D. Dolev and A. C. Yao. On the security of public key protocols. *Information Theory, IEEE Transactions on*, 29(2):198–208, 1983. 41, 43, 101, 122, 148

[88] S. Drimer, S. J. Murdoch, and R. Anderson. Thinking inside the box: system-level failures of tamper proofing. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pages 281–295. IEEE, 2008. 25

[89] S. Drimer, S. J. Murdoch, et al. Keep your enemies close: distance bounding against smartcard relay attacks. In *USENIX Security*, volume 2007, 2007. 25, 107

[90] M. Dworkin. *Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC*. US Department of Commerce, National Institute of Standards and Technology, 2007. 58, 68

[91] M. Emms, B. Arief, L. Freitas, J. Hannon, and A. van Moorsel. Harvesting high value foreign currency transactions from EMV contactless credit cards without the PIN. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 716–726. ACM, 2014. 107

[92] N. Ferguson. Single term off-line coins. In T. Helleseth, editor, *Advances in Cryptology EUROCRYPT 93*, volume 765 of *Lecture Notes in Computer Science*, pages 318–328. Springer Berlin Heidelberg, 1994. 38

[93] Financial Fraud Action UK. Fraud the facts 2017: The definitive overview of payment industry fraud. [Online]. Available: https://www.financialfraudaction.org.uk/fraudfacts17/assets/fraud_the_facts.pdf, 2017. Last visited on 08/04/2018. 28, 29, 47, 55

[94] P. FIPS. 180-4-federal information processing standards publication-secure hash standard (shs)-national institute of standards and technology gaithersburg, 2012. 77

[95] L. Francis, G. Hancke, K. Mayes, and K. Markantonakis. Practical NFC peer-to-peer relay attack using mobile phones. In *Proceedings of the 6th International Conference on Radio Frequency Identification: Security and Privacy Issues*, RFIDSec'10, pages 35–49, Berlin, Heidelberg, 2010. Springer-Verlag. 100

[96] Gautham. BTC.com Wallet App. [Online]. Available: http://www.newsbtc.com/2016/09/19/btc-com-wallet-app-sms-bitcoin/, September 2016. Last visited on 08/04/2018. 37

[97] S. Gibbs. SS7 hack explained: what can you do about it? [Online]. Available: https://www.theguardian.com/technology/2016/apr/19/ss7-hack-

explained-mobile-phone-vulnerability-snooping-texts-calls, April 2016. Last visited on 08/04/2018. 168

[98] D. Gilman. Cyber-Warfare and Humanitarian Space. [Online]. Available: http://commstech-hub.eisf.eu/uploads/4/0/2/4/40242315/daniel_gilman_cyberwarfare_and_humanitarian_space_eisf_october_2014.pdf, October 2014. Last visited on 08/04/2018. 159

[99] I. Gurulian, C. Shepherd, E. Frank, K. Markantonakis, R. N. Akram, and K. Mayes. On the effectiveness of ambient sensing for detecting NFC relay attacks. In *Trustcom/BigDataSE/ICESS, 2017 IEEE*, pages 41–49. IEEE, 2017. 100

[100] T. Halevi, D. Ma, N. Saxena, and T. Xiang. Secure proximity detection for NFC devices based on ambient sensor data. In *Computer Security, ESORICS 2012*, volume 7459 of *Lecture Notes in Computer Science*, pages 379–396. Springer Berlin Heidelberg, 2012. 100

[101] N. Haller, C. Metz, P. Nesser, and M. Straw. A one time password system RFC 2289. *Internet Engineering Task Force.*, 1998. 161, 162

[102] G. P. Hancke. Practical attacks on proximity identification systems. In *Security and Privacy, 2006 IEEE Symposium on*, pages 6–pp. IEEE, 2006. 100

[103] G. P. Hancke, K. Mayes, and K. Markantonakis. Confidence in smart token proximity: Relay attacks revisited. *Computers & Security*, 28(7):615–627, 2009. 100

[104] G. Hileman and M. Rauchs. Global cryptocurrency benchmarking study. *Cambridge Centre for Alternative Finance*, 2017. 33

[105] C. A. R. Hoare. *Communicating sequential processes*, volume 21. ACM, New York, NY, USA, 1978. 41, 69

[106] Q. Huang, D. Wong, and W. Susilo. The construction of ambiguous optimistic fair exchange from designated confirmer signature without random oracles. In M. Fischlin, J. Buchmann, and M. Manulis, editors, *Public Key Cryptography PKC 2012*, volume 7293 of *Lecture Notes in Computer Science*, pages 120–137. Springer Berlin Heidelberg, 2012. 40

[107] S. J. Hughes and S. T. Middlebrook. Regulating cryptocurrencies in the United States: Current issues and future directions. 2014. 127

[108] ImpACT Coalition. Through a glass DARLKY: The case for accelerating the drive for accountability, clarity and transparency in the charity sector. Technical report, 2013. 150, 152

[109] R. Indrakshi and R. Indrajit. An optimistic fair exchange e-commerce protocol with automated dispute resolution. In K. Bauknecht, S. Madria, and G. Pernul, editors, *Electronic Commerce and Web Technologies*, volume 1875 of *Lecture Notes in Computer Science*, pages 84–93. Springer Berlin Heidelberg, 2000. 39

[110] Information Security Media Group (ISMG). Global payments breach tab: $94 million. [Online]. Available: http://www.bankinfosecurity.com/global-payments-breach-tab-94-million-a-5415, 10th January 2012. Last visited on 08/04/2018. 53, 73

[111] International Telecommunications Union (ITU). ICT Facts And Figures 2016. [Online]. Available: http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2016.pdf, 2016. Last visited on 08/04/2018. 158

[112] J. P. Jalil, T. F. Zych, and E. M. Little. Why not accept Bitcoin for goods and services?, Corporate Counsel, 2014. 127

[113] D. Jayasinghe. Digital cash and anonymous fair-exchange payment protocols. *Technical Report*, 2016. 33, 34, 37

[114] D. Jayasinghe, R. N. Akram, K. Markantonakis, K. Rantos, and K. Mayes. Enhancing EMV online PIN verification. In *Trustcom/BigDataSE/ISPA, 2015 IEEE*, volume 1, pages 808–817, Aug 2015. 31, 107

[115] Joan Daemen and Vincent Rijmen. *The design of Rijndael: AES - the Advanced Encryption Standard*. Springer Verlag, Berlin, Heidelberg, New York, 2002. 58, 59, 67, 68

[116] B. Kaliski. Public Key Cryptography Standards (PKCS)#1: RSA encryption version 1.5. 1998. 61

[117] Kaspersky Lab. Equation group: the crown creator of cyber-espionage. [Online]. Available: http://www.kaspersky.com/about/news/virus/2015/equation-group-the-crown-creator-of-cyber-espionage, 16th February 2015. Last visited on 08/04/2018. 51, 53, 73, 107

[118] Kaspersky Lab. The great bank robbery. [Online]. Available: http://www.kaspersky.com/about/news/virus/2015/Carbanak-cybergang-

steals-1-bn-USD-from-100-financial-institutions-worldwide, 16th February 2015. Last visited on 08/04/2018. 51, 53, 73, 107

[119] J. P. Katoen, M. Khattri, and I. S. Zapreevt. A markov reward model checker. In *Second International Conference on the Quantitative Evaluation of Systems (QEST'05)*, pages 243–244, Sept 2005. 41

[120] J.-P. Katoen, I. S. Zapreev, E. M. Hahn, H. Hermanns, and D. N. Jansen. The ins and outs of the probabilistic model checker mrmc. *Performance Evaluation*, 68(2):90 – 104, 2011. Advances in Quantitative Evaluation of Systems. 41

[121] J. Kelsey, B. Schneier, and D. Wagner. *Advances in Cryptology — CRYPTO '96: 16th Annual International Cryptology Conference Santa Barbara, California, USA August 18–22, 1996 Proceedings*, chapter Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES, pages 237–251. Springer Berlin Heidelberg, 1996. 103

[122] Z. Kfir and A. Wool. Picking virtual pockets using relay attacks on contactless smartcard. In *Security and Privacy for Emerging Areas in Communications Networks, 2005. SecureComm 2005. First International Conference on*, pages 47–58. IEEE, 2005. 100

[123] D. King. Chip-and-PIN: Success and challenges in reducing fraud. In *Retail Payments Risk Forum, January*, 2012. 26

[124] P. Koshy, D. Koshy, and P. McDaniel. An analysis of anonymity in Bitcoin using p2p network traffic. In *International Conference on Financial Cryptography and Data Security*, pages 469–485. Springer, 2014. 37

[125] Krebs on Security. Payments Giant Verifone Investigating Breach. [Online]. Available: https://krebsonsecurity.com/2017/03/payments-giant-verifone-investigating-breach/, 7th March 2017. Last visited on 08/04/2018. 53, 73

[126] M. Kwiatkowska, G. Norman, and D. Parker. Prism: Probabilistic symbolic model checker. In T. Field, P. G. Harrison, J. Bradley, and U. Harder, editors, *Computer Performance Evaluation: Modelling Techniques and Tools*, pages 200–204, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg. 41

[127] M. Kwiatkowska, G. Norman, and D. Parker. Advances and challenges of probabilistic model checking. In *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1691–1698, Sept 2010. 41

[128] M. Kwiatkowska, G. Norman, and D. Parker. Prism 4.0: Verification of probabilistic real-time systems. In G. Gopalakrishnan and S. Qadeer, editors, *Computer Aided Verification*, pages 585–591, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. 41

[129] G. Lowe. Casper: a compiler for the analysis of security protocols. *J. Comput. Secur.*, 6:53–84, January 1998. 41, 69

[130] N. Mankiw. *Principles of microeconomics*, volume 10. Cengage Learning, 2006. 151

[131] M. Mannan and P. C. van Oorschot. Weighing down "the unbearable lightness of PIN cracking". In *Financial Cryptography and Data Security*, pages 176–181. Springer, 2008. 26, 50

[132] MasterCard. Mondex. [Online]. Available: `http://www.mondexusa.com/`. Last visited on 08/04/2018. 38

[133] K. Mayes and K. Markantonakis, editors. *Smart Cards, Tokens, Security and Applications*. Second Edition. Springer, 2017. 25, 47, 60, 76

[134] Mayes, Keith and Markantonakis, Konstantinos. *Smart cards, tokens, security and applications*. Springer Science and Business Media, 2007. 161, 162

[135] D. McGrew and J. Viega. The Galois/counter mode of operation (GCM). [Online]. Available: `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/gcm/gcm-spec.pdf`, 2004. Last visited on 08/04/2018. 58, 68

[136] M. Mehrnezhad, F. Hao, and S. F. Shahandashti. *Tap-Tap and Pay (TTP): Preventing Man-In-The-Middle Attacks in NFC Payment Using Mobile Sensors*. Computing Science, Newcastle University, 2014. 100

[137] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 127–140. ACM, 2013. 141, 147

[138] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC, October 1996. 67, 85

[139] R. C. Merkle. Protocols for public key cryptosystems. In *IEEE Symposium on Security and Privacy*, pages 122–134, 1980. 36

[140] T. Merschen. Fraud dynamics in the card payments industry: A global review of the realities of EMV deployment. *Journal of Payments Strategy & Systems*, 4(2):156–169, 2010. 29, 47

[141] I. Miers, C. Garman, M. Green, and A. Rubin. Zerocoin: Anonymous distributed e-cash from bitcoin. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 397–411, 2013. 37, 141, 143

[142] D. Mills, K. Wang, B. Malone, A. Ravi, J. Marquardt, C. Chen, A. Badev, T. Brezinski, L. Fahy, K. Liao, et al. Distributed ledger technology in payments, clearing, and settlement. [Online]. Available: `https://www.federalreserve.gov/econresdata/feds/2016/files/2016095pap.pdf`. Last visited on 08/04/2018. 33

[143] D. M'Raihi, S. Machani, M. Pei, and J. Rydell. Totp: Time-based one-time password algorithm. Technical report, 2011. 161, 162

[144] P. C. Mullan. *Disadvantages and Barriers*, pages 115–117. Palgrave Macmillan US, New York, 2014. 127

[145] C. Mulliner, N. Golde, and J.-P. Seifert. SMS of death: From analyzing to attacking mobile phones on a large scale. In *USENIX Security Symposium*, 2011. 168

[146] S. J. Murdoch, S. Drimer, R. Anderson, and M. Bond. Chip and PIN is Broken. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 433–446. IEEE, 2010. 25, 67, 85

[147] S. Nakamoto. Bitcoin: A peer-to-peer e-cash system. [Online]. Available: `http://www.bitcoin.org/bitcoin.pdf`, 2008. Last visited on 08/04/2018. 33, 34, 35, 36, 153, 172

[148] A. Nandwani. Coinbase SpringCleaning. [Online]. Available: `https://blog.coinbase.com/coinbase-spring-cleaning-4f27710ff821`, March 2017. Last visited on 08/04/2018. 37

[149] NIST, US. Descriptions of SHA-256, SHA-384 and SHA-512, 2001. 58

[150] O. Ogundele, P. Zavarsky, R. Ruhl, and D. Lindskog. Fraud reduction on emv payment cards by the implementation of stringent security features. *International Journal of Intelligent Computing Research (IJICR)*, 3(1/2):252–262, 2012. 28, 29, 47

[151] T. Okamoto and K. Ohta. Electronic digital cash. In *Advances in Cryptology;CRYPTO 91*, pages pp 324–350, Berlin, 1991. Springer. 36

[152] D. Ortiz-Yepes. A critical review of the EMV payment tokenisation specification. *Computer Fraud & Security*, (10):5 – 12, 2014. 104

[153] J. H. Park and J. H. Park. Blockchain security in cloud computing: Use cases, challenges, and solutions. *Symmetry (20738994)*, 9(8), 2017. 172

[154] PayPal. CyberCash. [Online]. Available: https://www.paypal-knowledge.com/infocenter/index?page=content&id=FAQ1761&actp=RSS. Last visited on 24/02/2016. 38

[155] PCI Security Standards Council. Information supplement: PCI DSS tokenization guidelines, Version 2.0, PCI Data Security Standard (PCI DSS), Auguest 2011. 30

[156] M. Pisa and M. Juden. Blockchain and economic development: Hype vs. reality. *Center for Global Development Policy Paper*, 107, 2017. 150

[157] Populus. Public trust and confidence in charities. [Online]. Available: https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/532104/Public_trust_and_confidence_in_charities_2016.pdf, 2016. Last visited on 08/04/2018. 150, 179

[158] P. Pourghomi, P. E. Abi-Char, and G. Ghinea. Towards a mobile payment market: A comparative analysis of host card emulation and secure element. *International Journal of Computer Science and Information Security*, 13(12):156, 2015. 92

[159] W. Rankl and W. Effing. *Smart Card Handbook*. John Wiley & Sons, Inc., New York, NY, USA, third edition, 2003. 25, 47

[160] W. Rankl and W. Effing. *Smart card handbook*. John Wiley and Sons, 2010. 102, 103

[161] I. Ray and I. Ray. An anonymous fair exchange e-commerce protocol. In *In Proceedings of the International Workshop on Internet Computing and ECommerce*, pages 172–179. IEEE Computer Society, 2001. 126

[162] I. Ray and I. Ray. Fair exchange in e-commerce. *SIGecom Exch.*, 3(2):9–17, Mar. 2002. 38, 39

[163] I. Ray, I. Ray, and N. Natarajan. An anonymous and failure resilient fair-exchange e-commerce protocol. *Decis. Support Syst.*, 39(3):267–292, May 2005. 36

[164] F. Reid and M. Harrigan. An analysis of anonymity in the Bitcoin system. In Y. Altshuler, Y. Elovici, A. B. Cremers, N. Aharony, and A. Pentland, editors, *Security and Privacy in Social Networks*, pages 197–223. Springer New York, 2013. 37, 141, 147

[165] Reuters. Target in $18.5 million multi-state settlement over data breach. [Online]. Available: http://www.reuters.com/article/us-target-cyber-settlement-idUSKBN18J2GH, September 2017. Last visited on 08/04/2018. 29, 30

[166] RFC 4226. HOTP: An HMAC-based one-time password algorithm. [Online]. Available: http://www.ietf.org/rfc/rfc4226.txt, December 2005. Last visited on 08/04/2018. 163

[167] R. Rivest. The MD5 message-digest algorithm. 1992. 102

[168] B. Rodrigues, T. Bocek, A. Lareida, D. Hausheer, S. Rafati, and B. Stiller. A Blockchain-based architecture for collaborative DDoS mitigation with smart contracts. In *IFIP International Conference on Autonomous Infrastructure, Management and Security*, pages 16–29. Springer, 2017. 172

[169] M. Roland, J. Langer, and J. Scharinger. Relay attacks on secure element-enabled mobile devices. In *Information Security and Privacy Research*, volume 376, pages 1–12. Springer Berlin Heidelberg, 2012. 100

[170] D. Ron and A. Shamir. Quantitative analysis of the full bitcoin transaction graph. In *International Conference on Financial Cryptography and Data Security*, pages 6–24. Springer, 2013. 37

[171] A. W. Roscoe. A Classical Mind. http://dl.acm.org/citation.cfm?id=197600.197628, 1994. 41

[172] P. Ryan and S. Schneider. *The Modelling and Analysis of Security Protocols: the CSP Approach.* Addison-Wesley Professional, 2000. 69

[173] Safaricom. M-PESA. [Online]. Available: https://www.safaricom.co.ke/personal/m-pesa, 2016. Last visited on 08/04/2018. 37

[174] I. Sakharova. Payment card fraud: Challenges and solutions. In *Intelligence and Security Informatics (ISI), 2012 IEEE International Conference on*, pages 227–234. IEEE, 2012. 29, 47

[175] A. Scannell, A. Varshavsky, A. LaMarca, and E. De Lara. Proximity-based authentication of mobile devices. *International Journal of Security and Networks*, 4(1-2):4–16, 2009. 100

[176] B. Schneier. *Applied cryptography (2nd ed.): protocols, algorithms, and source code in C*. John Wiley & Sons, Inc., New York, NY, USA, 1995. 61, 68, 102

[177] R. Schollmeier. A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In *Peer-to-Peer Computing, 2001. Proceedings. First International Conference on*, pages 101–102. IEEE, 2001. 33

[178] B. Shrestha, N. Saxena, H. T. T. Truong, and N. Asokan. Drone to the rescue: Relay-resilient authentication using ambient multi-sensing. In *Financial Cryptography and Data Security*, pages 349–364. Springer, 2014. 100, 101

[179] X. Shu, K. Tian, A. Ciambrone, et al. Breaking the target: an analysis of target data breach and lessons learned. *arXiv preprint arXiv:1701.04940*, 2017. 29, 30

[180] Smart Card Alliance. Host Card Emulation (HCE) 101. *A Smart Card Alliance Mobile & NFC Council White Paper*, 2014. 92

[181] Smart Card Alliance. The true cost of data breaches in the payments industry. Technical report, March 2015. 29, 47

[182] R. J. Sullivan. Can smart cards reduce payments fraud and identity theft? *Federal Reserve Bank of Kansas City, Economic Review*, 93(3):35–62, 2008. 26

[183] Symantec. A special report on: attacks on point-of-sales systems. November 2014. 107

[184] The Charity Commission. Charity Commission annual report and accounts 2015 to 2016. [Online]. Available: `https://www.gov.uk/government/publications/charity-commission-annual-report-and-accounts-2015-to-2016`, 2016. Last visited on 08/04/2018. 150

[185] The Charity Commission. Compliance toolkit protecting charities from harm: Due diligence, monitoring and verification of end use of charitable funds. [Online]. Available: `http://forms.charitycommission.gov.uk/media/89350/compliance_toolkit_2.pdf`, 2016. Last visited on 08/04/2018. 170

[186] The Charity Commission. New charity investigation: Kids company. [Online]. Available: https://www.gov.uk/government/news/new-charity-investigation-kids-company, 2016. Last visited on 08/04/2018. 150, 179

[187] The UK Cards Association. Interchange. [Online]. Available: http://www.theukcardsassociation.org.uk/Interchange/index.asp, 2016. Last visited on 08/04/2018. 152

[188] The World Bank. Remittance prices worldwide. [Online]. Available: https://remittanceprices.worldbank.org/sites/default/files/rpw_report_sept_2016.pdf, 2016. Last visited on 08/04/2018. 153

[189] The World Bank. Remittance prices worldwide: Third quarter 2016. [Online]. Available: https://remittanceprices.worldbank.org/en, 2016. Last visited on 08/04/2018. 153

[190] The World Bank Group. The global Findex database 2014: Measuring financial inclusion around the world. Technical report, 2014. 151

[191] A. Umar, K. Mayes, and K. Markantonakis. Performance variation in host-based card emulation compared to a hardware security element. In *Mobile and Secure Services (MOBISECSERV), 2015 First Conference on*, pages 1–6. IEEE, 2015. 102

[192] U.N. Secretary-General Ban Ki-moon. Secretary-general's closing remarks at high-level panel on accountability, transparency and sustainable development. Technical report, United Nations, 2012. 150

[193] P. Urien. Collaboration of SSL smart cards within the WEB2 landscape. 2009. 103

[194] P. Urien and S. Elrharbi. Tandem smart cards: enforcing trust for tls-based network services. In *Applications and Services in Wireless Networks, 2008. ASWN'08. Eighth International Workshop on*, pages 96–104. IEEE, 2008. 103

[195] Visa Public . *Issuer PIN Security Guidelines*, November 2010. 26, 50, 65, 76

[196] M. Walport. Distributed ledger technology: Beyond blockchain. *UK Government Office for Science, Tech. Rep*, 19, 2016. 33, 150

[197] W. Weng, C. Woo, Y. Cheng, T. Ho, and I. Horowitz. Public trust and corruption perception: disaster relief. *Applied Economics*, 47(46):4967–4981, 2015. 150

[198] J. R. Willett. Mastercoin complete specification v1.0. [Online]. Available: https://github.com/mastercoin-MSC/spec, January 2014. Last visited on 24/02/2016. 37

[199] G. Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 2014. 150

[200] WorldPay. Optimising your alternative payments: Global view. Whitepaper, 2010. 125

[201] WorldTimeZone.com. GSM world coverage map and GSM country list. [Online]. Available: http://www.worldtimezone.com/gsm.html. Last visited on 08/04/2018. 158

[202] A. Wright and P. De Filippi. Decentralized blockchain technology and the rise of lex cryptographia. *Available at SSRN 2580664*, 2015. 33, 150, 153

[203] A. Yelowitz and M. Wilson. Characteristics of Bitcoin users: an analysis of Google search data. *Applied Economics Letters*, 22(13):1030–1036, 2015. 37

[204] D. Yermack. Is Bitcoin a real currency? an economic appraisal. Technical report, National Bureau of Economic Research, 2013. 127

[205] J. Young. Former Kipochi CTO explains controversial m-pesa deal. [Online]. Available: http://www.newsbtc.com/2016/01/11/former-kipochi-ceo-explains-controversial-m-pesa-bitcoin-deal/, January 2016. Last visited on 08/04/2018. 37

[206] N. Zhang and Q. Shi. Achieving non-repudiation of receipt. *The Computer Journal*, 39(10):844–853, 1996. 40

[207] N. Zhang, Q. Shi, and M. Merabti. An efficient protocol for anonymous and fair document exchange. *Computer Networks*, 41(1):19 – 28, 2003. 39, 126

[208] Q. Zhang, K. Markantonakis, and K. Mayes. A mutual authentication enabled fair-exchange and anonymous e-payment protocol. *E-Commerce Technology, IEEE International Conference on, and Enterprise Computing, E-Commerce, and E-Services, IEEE International Conference on*, 0:20, 2006. 39, 126

[209] Q. Zhang, K. Mayes, and K. Markantonakis. *A user-centric m-payment solution*, page 8. 2005. 39, 40, 126

[210] J. Zhou and D. Gollman. A fair non-repudiation protocol. In *Security and Privacy, 1996. Proceedings., 1996 IEEE Symposium on*, pages 55–61, 1996. 40

Appendix A

Source Code - Formal Analysis Script

The scripts modeled for mechanical formal analysis in the thesis is included in Appendix A.

A.1 CasperFDR Script in Chapter 3

The protocols proposed in Chapter 3 are modeled as follows:

A.1.1 Symmetric System

```
#Free variables
SC, CI: Agent
Dh, Pb, Rp, Cvr; Num
Cun, Aun: Nonce
InverseKeys = (EnMaKey, EnMaKey), (SessionEnMaKey, SessionEnMaKey)

#Protocol description
0.    -> SC : SC [CI!=SC] <iMsg :={Dh, Pb, Cun,SessionEnMaKey,Rp}{EnMaKey}>
1. SC -> CI : SC, iMsg
2. CI -> SC : {DH,Cvr, Aun, Cun}{SessionEnMaKey}

#Actual variables
SCard, CIssuer, ME: Agent
DH, PB, RP, CVR: Num
CUN, AUN, NMalicious: Nonce

#Processes
INITIATOR(SC,CI, Cun) knows EnMaKey
RESPONDER(CI,SC, Aun) knows EnMaKey

#System
INITIATOR(SCard, SIssuer, CUN)
RESPONDER(SIssuer, SCard, AUN)

#Intruder Information
Intruder = ME
IntruderKnowledge = {SIssuer, SCard, ME,
GMalicious, NMalicious}
```

```
#Specification
Aliveness(SI, SC)
Aliveness(SC, SI)
Secret(SC, EnMaKey, [CI])
Secret(Sc, SessionEnMaKey, [CI])
```

A.1.2 Asymmetric System

```
#Free variables
SC, CI: Agent
Dh, Pb, Rp, Cvr; Num
Cun, Aun: Nonce
EKey: Agent->PublicKey
DKey: Agent->SecretKey
InverseKeys = (VKey, SKey), (SessionEnMaKey, SessionEnMaKey)

#Protocol description
0.    -> SC : SC [CI!=SC] <iMsg := {Dh, Pb, Cun, SessionEnMaKey, Rp}{EKey(CI)}>
1. SC -> CI : SC, iMsg
2. SC -> CI : {Dh, Cvr, Aun, Cun}{SessionEnMaKey}

#Actual variables
SCard, CIssuer, ME: Agent
DH, PB, RP, CVR: Num
CUN, AUN, NMalicious: Nonce

#Processes
INITIATOR(SC,CI, Cun)knows EKey
RESPONDER(SC,SI, Aun) knows DKey(CI), EKey

#System
INITIATOR(CIssuer, SCard, CUN)
RESPONDER(SCard, CIssuer, AUN)

#Functions
symbolic EKey, DKey

#Intruder Information
Intruder = ME
IntruderKnowledge = {CIssuer, SCard, ME, NMalicious, DKey(ME), EKey}

#Specification
Aliveness(CI, SC)
Aliveness(SC, CI)
Secret(SC, SessionEnMaKey, [CI])
```

A.2 Scyther Script in Chapter 5

The protocol proposed in Chapter 5 is modeled as follows:

```
usertype Data;
hashfunction h;
usertype SessionKey;
const Cert: Function;
secret Cert1: Function;

protocol ot2(SE,T)
{
role SE {
fresh nse: Nonce;  fresh PDOL: Data;
var nt: Nonce;
fresh OTT: Data;  fresh TSPsig: Data;
macro DAD = nt2, OTT;
fresh K: SessionKey; fresh nse2: Nonce;
var X: Ticket;  var nt2: Nonce;
recv_1(T,SE, T,nt,Cert1(T));
send_2(SE,T, {SE,T,nse,nt,PDOL,K}pk(T));
recv_3(T,SE, {T,SE,nse,nt2,X}K);
send_4(SE,T, {SE,T,nse2,nt2,DAD,TSPsig}K, {h(DAD)}sk(SE),Cert(SE) );

claim(SE, Alive);
claim(SE, Secret, K);
claim(SE, Niagree);
claim(SE, Nisynch);
claim(SE, Secret, OTT);
claim(SE, Secret, X);
}

role T {
var nse: Nonce; var PDOL: Data;
fresh nt: Nonce;  fresh TTQ: Data;
fresh amount: Data; fresh nt2: Nonce;
fresh CurrencyCode: Data;
var K: SessionKey; var Y: Ticket;
macro m1 = amount,CurrencyCode;
var TSPsig: Data;  var nse2: Nonce;

send_1(T,SE, T,nt,Cert1(T));
recv_2(SE,T, {SE,T,nse,nt,PDOL,K}pk(T));
send_3(T,SE, {T,SE,nse,nt2,m1}K);
recv_4(SE,T, {SE,T,nse2,nt2,Y,TSPsig}K, {h(Y)}sk(SE),Cert(SE) );

claim(T, Alive);
claim(T, Secret, K);
claim(T, Niagree);
claim(T, Nisynch);
claim(T, Secret, Y);
}
}
```

A.3 Scyther Script in Chapter 6

The protocol proposed in Chapter 6 is modeled as follows:

```
usertype Data;
hashfunction h;
usertype SessionKey;
const Cert: Function;
secret Cert1: Function;

protocol ot1(SE,T,TSP)
{
role SE
{
fresh nse: Nonce;
var nt: Nonce;
fresh K: SessionKey;
var nt2: Nonce;
fresh nse2: Nonce;
var ntsp: Nonce;
var W: Ticket;
var Y: Ticket;
fresh nse3: Nonce;;

recv_1(T,SE, T,nt, Cert1(T));
send_2(SE,T, {SE,T,nse,nt, K,TSP}pk(T),{h({SE,T,nse,nt, K,TSP}pk(T))}sk(SE),Cert1(SE));
recv_3(T,SE, {T,SE,nse,nt2,W}K,{h({T,SE,nse,nt2,W}K)}sk(T));
send_4(SE,TSP, {SE,TSP, nse2,Cert1(T),W}k(SE,TSP));
recv_5(TSP,SE, {TSP,SE, nse2,ntsp,Y}k(SE,TSP));
send_6(SE,T, {SE,T, nse3,nt2,Y}K);
claim(SE, Alive);
claim(SE, Secret, K);
claim(SE, Niagree);
claim(SE, Nisynch);
claim(SE, Weakagree);
}

role T {
fresh nt: Nonce;
var nse: Nonce;
var K: SessionKey;
fresh nt2: Nonce;
fresh nt3: Nonce;
var nse3: Nonce;
var ntsp2: Nonce;
var Token: Nonce;
var DTT: Data;
var Ks2: SessionKey;
macro DTD = TSP,T,ntsp2,nt3,Token,DTT,Ks2;
fresh nt4: Nonce;
var ARC: Data;
var ntsp3: Nonce;

send_1(T,SE, T,nt, Cert1(T));
recv_2(SE,T, {SE,T,nse,nt, K,TSP}pk(T),{h({SE,T,nse,nt, K,TSP}pk(T))}sk(SE),Cert1(SE));
send_3(T,SE, {T,SE,nse,nt2, {T,TSP,nt3}pk(TSP)}K,{h({T,SE,nse,nt2,{T,TSP,nt3}pk(TSP)}K)}sk(T));
recv_6(SE,T, {SE,T, nse3,nt2,{DTD}pk(T),{h(DTD)}sk(TSP)}K);
send_7(T,TSP, {T,TSP,ntsp2,nt4,DTT}Ks2);
recv_8(TSP,T, {TSP,T, ntsp3,nt4,ARC}Ks2);

claim(T, Alive);
claim(T, Secret, K);
claim(T, Niagree);
claim(T, Nisynch);
claim(T, Secret, DTT);
claim(T, Weakagree);
}

role TSP {
var nse2: Nonce;
var nt3: Nonce;
fresh ntsp: Nonce;
fresh Token: Data;
fresh DTT: Data;
fresh Ks2: SessionKey;
fresh ntsp2: Nonce;
macro DTD = TSP,T,ntsp2,nt3,Token,DTT,Ks2;
var nt4: Nonce;
fresh ntsp3: Nonce;
fresh ARC: Data;

recv_4(SE,TSP, {SE,TSP, nse2,Cert1(T),{T,TSP,nt3}pk(TSP)}k(SE,TSP));
send_5(TSP,SE, {TSP,SE, nse2,ntsp,{DTD}pk(T),{h(DTD)}sk(TSP)}k(SE,TSP));
recv_7(T,TSP, {T,TSP,ntsp2,nt4,DTT}Ks2);
send_8(TSP,T, {TSP,T,ntsp3,nt4,ARC}Ks2);

claim(TSP, Alive);
claim(TSP, Secret, Ks2);
claim(TSP, Secret, ARC);
claim(TSP, Niagree);
claim(TSP, Nisynch);
claim(TSP, Weakagree);
}
}
```

A.4 Scyther Script in Chapter 7

The proposed main and extended protocol stages in Chapter 7 are modeled as follows:

A.4.1 Main Fair-exchange Protocol

```
usertype Data;
hashfunction h;
usertype SessionKey;
const Cert: Function;
secret Cert1: Function;

protocol fe(M,C)
{
role M
{
fresh nm1: Nonce;
fresh Pid: Data;
fresh Invoice: Data;
fresh PVc: Data;
fresh TTPcom: Data;
fresh t: Data;
fresh m: Data;
fresh K: SessionKey;
macro m1 = Pid,Invoice,{h(Invoice)}sk(M),nm1,PVc,{m}K,TTPcom,Cert1(M),t;
var nc: Nonce;
var tp: Data;
macro m2 = Invoice,{h(Invoice)}sk(M),{h({h(Invoice)}sk(M))}sk(C),nm1,nc,h({m}K),tp;
fresh nm2: Nonce;
macro m3 = Invoice,nc,nm2,h({m}K),K,t;

send_1(M,C, {m1}pk(C), {h({m1}pk(C))}sk(M));
recv_2(C,M, {m2}pk(M),{h({m2}pk(M))}sk(C));
send_3(M,C, {m3}pk(C), {h({m3}pk(C))}sk(M));

claim(M, Alive);
claim(M, Secret, K);
claim(M, Secret, m);
claim(M, Niagree);
claim(M, Nisynch);
claim(M, Weakagree);
}

role C
{
var nm1: Nonce;
var Pid: Data;
var PVc: Data;
var Invoice: Data;
var TTPcom: Data;
var m: Data;
var t: Data;
macro m1 = Pid,Invoice,{h(Invoice)}sk(M),nm1,PVc,{m}K,TTPcom,Cert1(M),t;
fresh nc: Nonce;
var m: Data;
fresh tp: Data;
macro m2 = Invoice,{h(Invoice)}sk(M),{h({h(Invoice)}sk(M))}sk(C),nm1,nc,h({m}K),tp;
var K: SessionKey;
var nm2: Nonce;
macro m3 = Invoice,nc,nm2,h({m}K),K,t;

recv_1(M,C, {m1}pk(C),{h({m1}pk(C))}sk(M));
send_2(C,M, {m2}pk(M),{h({m2}pk(M))}sk(C));
recv_3(M,C, {m3}pk(C), {h({m3}pk(C))}sk(M));

claim(C, Alive);
claim(C, Secret, K);
claim(C, Secret, m);
claim(C, Niagree);
claim(C, Nisynch);
claim(C, Weakagree);
}
}
```

A.4.2 Extended Fair-exchange Protocol

```
usertype Data;
hashfunction h;
usertype SessionKey;
const Cert: Function;
secret Cert1: Function;

protocol fe(M,C,TTP)
{
role C
{
var nm0: Nonce;
var m: Data;
macro m0 = {m}K, nm0;
fresh nc1: Nonce;
fresh t: Data;
fresh BE: Data;
fresh Invoice: Data;
fresh TTPcom: Data;
fresh m: Data;
macro m1 = BE,Invoice,{h(Invoice)}sk(M),nc1,h({m}K),TTPcom,t;
var K: SessionKey;
var nttp2: Nonce;
var nc1: Nonce;
macro m4 = Invoice,K,nttp2,nc1,t;

recv_1(M,C, {m0}pk(C), {h({m0}pk(C))}sk(M));
send_2(C,TTP, {m1}pk(TTP),{h({m1}pk(TTP))}sk(M));
recv_5(TTP,C, {m4}pk(C), {h({m4}pk(C))}sk(TTP));

claim(C, Alive);
claim(C, Secret, K);
claim(C, Secret, m);
claim(C, Niagree);
claim(C, Nisynch);
claim(C, Weakagree);
}

role TTP
{
var nc1: Nonce;
var t: Data;
var Invoice: Data;
var TTPcom: Data;
var m: Data;
var BE: Data;
macro m1 = BE,Invoice,{h(Invoice)}sk(M),nc1,h({m}K),TTPcom,t;
fresh nttp1: Nonce;
fresh KR: Data;
fresh tr: Data;
macro m2 = BE,Invoice,{h(Invoice)}sk(M),KR,nttp1,tr;
var K: SessionKey;
var nm1: Nonce;
macro m3 = Invoice,K,nttp1,nm1,t;
fresh nttp2: Nonce;
macro m4 = Invoice,K,nttp2,nc1,t;

recv_2(C,TTP, {m1}pk(TTP),{h({m1}pk(TTP))}sk(M));
send_3(TTP,M, {m2}pk(M),{h({m2}pk(M))}sk(TTP));
recv_4(M,TTP, {m3}pk(TTP), {h({m3}pk(TTP))}sk(M));
send_5(TTP,C, {m4}pk(C), {h({m4}pk(C))}sk(TTP));
claim(TTP, Alive);
claim(TTP, Secret, K);
claim(TTP, Secret, m);
claim(TTP, Niagree);
claim(TTP, Nisynch);
claim(TTP, Weakagree);
}
}

role M
{
fresh nm0: Nonce;
fresh m: Data;
fresh K: SessionKey;
macro m0 = {m}K, nm0;
macro m2 = BE,Invoice,{h(Invoice)}sk(M),KR,nttp1,tr;
var BE: Data;
var Invoice: Data;
var KR: Data;
var nttp1: Nonce;
var tr: Data;
fresh nm1: Nonce;
fresh t: Data;
macro m3 = Invoice,K,nttp1,nm1,t;

send_1(M,C, {m0}pk(C), {h({m0}pk(C))}sk(M));
recv_3(TTP,M, {m2}pk(M),{h({m2}pk(M))}sk(TTP));
send_4(M,TTP, {m3}pk(TTP), {h({m3}pk(TTP))}sk(M));

claim(M, Alive);
claim(M, Secret, K);
claim(M, Secret, m);
claim(M, Niagree);
claim(M, Nisynch);
claim(M, Weakagree);
}
}
```