ABSTRACT

For the growing use of counterfeit cards frauds have been investigated during the transactions in the past decades, which led the banking industry to be threatened and transformed from the traditional magnetic-stripe-based card to chip-based smart card technology, giving birth to the global EMV standards for bank cards transactions to enforce the payment security. Formally, EMV is known as an open-standard set of specifications for EMV-based payment cards and card accepting devices. It can provide security and global interoperability by denying not only how smart bank cards can be used to prevent counterfeit fraud but also how they can be used for multiple payment applications on the identical cards. [1, 3] Although the researchers and the banking industry have made their joint effort to enhance the security of EMV-based smart card for payment systems, with the extensive use of EMV-based smart cards, an important issue that arise in the payment systems is to protect all involved participants from being attacked to secure transactions. This has led to an interest towards developing new security deterrence and protection mechanisms. More formally, the current infrastructure of EMV-based payment systems has been suggested to work properly to provide the expected security for the transaction data, however it has also been examined that it is inefficient to resist certain attacks during the transactions.[2] Hence, developing a security model for EMV-based payment transactions is getting necessary and increasing attractions for the research area, although the expectations of developing a full security model for EMV would be unrealistic and may be achievable in the coming future.

Based on these facts, this dissertation deploy a research methodology that investigate the security strategies followed by the EMV specifications, with the purpose to understand the vulnerabilities of the EMV transactions and to design a theoretical game-based security model for EMV. To conclude with discussing the detailed specifications of the generic construction from the aspect of cryptographic primitives.

Key Words: EMV specifications, security requirements, vulnerabilities, security model, cryptographic primitives.

ACKNOWLEDGEMENTS

I would like to express my great gratitude to my supervisor **Pro. Nigel Smart**, initially for his acquiescence to proceed with this dissertation throughout this project, as well as his continuous support and enthusiasm during each stage. And I deeply appreciate the help of Nigel for helping me organizing the final text and writing this dissertation professionally.

Finally, I would also like to thank my friends and my family for their moral support and help on several issues throughout this whole year.

This dissertation is dedicated to my parents.

DECLARATION

A dissertation submitted to the University of Bristol in
accordance with the requirements of the degree of Master of Science in
the Faculty of Engineering. It has not been submitted for any other
degree or diploma of any examining body. Except where
specifically acknowledged, it is all the work of the Author.

Xiaochen Fu, September 2011

Contents

LIST OF FIGURES

LIST OF ABBREVIATIONS

AAC	Application Authentication Cryptogram
AC	Application Cryptogram
AFL	Application File Locator
AIP	Application Interchange Profile
ARPC	Authorisation Response Cryptogram
ARQC	Authorisation Request Cryptogram
ATC	Application Transaction Counter
CA	Certification Authority
CAM	Card Authentication Method
CAP	Chip Authentication Programme
CDA	Combined DDA and application cryptogram generation
CDOL	Card Risk Management DOL
$Cert_I$	Certification for Issuer’s P_I
$Cert_{IC}$	Certification for IC cards’ P_{IC}
CID	Cryptogram Information Data
CNP	Card NOT Present
CVM	Cardholder Verification Method
CVR	Card Verification Rules
DDA	Dynamic Data Authentication
DDOL	Dynamic Data Authentication DOL
DOL	Data Object List
EMV	Europay Mastercard Visa

IAC	INTERNAL_AUTHENTICATE command
IAD	Issuer Application Data
ICC	Integrated Circuit Card
IKE	Internet Key Exchange
MAC	Message Authentication Code
MK_I	Master Key for Issuer
MK_{IC}	Master Key for IC cards
(P_{CA}, S_{CA}**)**	CA public key
(P_I, S_I**)**	RSA public-private key pair for Issuer
(P_{IC}, S_{IC}**)**	RSA public-private key pair for IC cards
(P_{SE}, S_{SE}**)**	public-private key pair for PIN Encipherment/Decipherment
PAN	Personal Account Number
PANs	PAN sequence number
PDOL	Processing Options DOL
PI	Purchase Information
PTC	PIN Entry Counter
PIN	Personal Identification Number
PKI	Public Key Infrastructure
POS	Point of Sale
SDA	Static Data Authentication
SDAD	Signed Dynamic Application Data
SK	Session Key
SSL	Secure Socket Layer
SSAD	Signed Static Application Data
TC	Transaction Certificate
TLS	Transport Layer Security

SUMMARY

Basically, this dissertation presents a comprehensive research on the security of EMV specifications and provides a theoretical cryptographic model for securing EMV transactions. Initially, it provided an overview of the EMV specifications by outlining the use of EMV for securing the communication between the involved parties, which were designed and introduced to effectively prevent certain attacks, such as fraud occurring in traditional magnetic stripe cards. Based on the knowledge of the fundamentals and the functionalities for EMV specifications, the dissertation then came up with the security requirements for the EMV specifications from a general level. Concurrently, it also captured the adversary's capability to act as a primer for the adversary's game specified in the construction. As a result, a theoretical game-based cryptographic security model for EMV specification was constructed by giving the formal syntax and algorithms along with the oracles that can be queried by the adversary in the format of the pseudo-code. To conclude this dissertation, the proofs for the modeling results are roughly given as one would expect and then ends with an overview and the future direction for the whole project.

The motivation of this project is that the proliferation of EMV cards and information technique has not only provided the general public with more convenient for their daily transactions, but also resulted in an increasing number of challenge for the security of banking transactions. Although the researchers and the banking industry have made their joint effort to enhance the security of EMV-based smart card for payment systems[3], with the extensive use of EMV-based smart cards, an important issue that arise in the payment systems is to protect all involved participants from being attacked to secure transactions. Hence, the aim and objective of this project is to develop a cryptographic security model for EMV, which would involve with extensive and suitable quality of literature review (nearly 60 bibliographies are referred to) to help seek solutions to secure transactions. It is believed that given the large complexity of the EMV specifications and the comprehensive cryptographic techniques, the constructed security model for EMV can be a useful basis for understanding the security requirements of the EMV protocol suite.

Finally, the main achievements and contributions can be concluded as following:

- This dissertation presents a comprehensive literature research on the EMV specifications, which helps direct the dissertation to reach the desired security outcomes; (see chapter 2: an overview of theory)
- The analysis of formal syntax and algorithms for the constructed security model are described as a pseudo-code based on the study of [8], which covers all the important options for the EMV cards and the Merchant Terminal, and the adversary's power is also modeled, see pages 28-34 and 37-39;
- It goes beyond the aim and objective what was set for the dissertation, which researching and proving the security requirements of our generic construction by the assumption of the involved cryptographic schemes secure against the corresponding attack, see pages 45-53.

1 Introduction

1.1 Background of the Project

As the booming of information technology, e-Banking and e-Commence have experienced an unprecedented and extraordinary transformation in recent decades, one of the significant change is that the integration of information technique and e-Commence (e-Banking) has provided the general public with more convenient for their daily transactions. Looking back into the history, card transactions were more physical than digital in the early 1970s, which magnetic stripe cards were widely used at that time. However, that has not last too long. For we are stepping into an increasingly interconnected world, the development of e-Banking and e-Commence has also resulted in an increasing number of counterfeit cards frauds, which resulted in the financial industry to be threatened and transformed from the traditional magnetic stripe card to chip-based smart card technology, giving birth to the global EMV standards for bank cards transactions.

Basically, the global EMV standards, a representative example of an exsiting payment smart-card specification, define the processing of bank credit and debit card payments at multiple levels (physical, electrical, data and application level) between a chip-based payment card and the processing payment terminal for fianncial transactions. Ever since the advantages of chip-based smart cards over magnetic stripe cards has been recognized by a great number of banks and other major financial organisations throughout the world, it is believed that chip-based smart cards would play a significant role in the coming future. The reason is that smart card technology embeds a secure integrated circuit chip with a microprocessor into their payment card, which contains the information needed to use the card for payment and a multitude of protection features, making it significantly more secure than a traditional magnetic stripe card. [1]

In general, these transactions secured by EMV specifications are also referred to ”Chip and PIN”-based, especially in the UK where chip-based EMV cards are highly advertised as ”Chip and PIN” cards. In addition, ”Chip and PIN” is intended to reduce fraud by requiring the genuine card and matching PIN be presented for a successful transaction. For instance, instead of signing a paper receipt to verify a card payment, the EMV Specifications enable customers to authorize a transaction by inserting their smart cards into a processing device and entering a PIN from the PIN pad to wait their smart cards being verified under this PIN entry, and then a public-key certificate authenticates it to the device. Moreover, the card issuer might further authenticate transactions online.[2]

It is clear that EMV is the abbreviation of Europay, MasterCard and Visa, first available in 1996 and then managed by EMVCo, which defines the global interoperable standard on how smart cards used for payment system. It is commonly known as the chip-based card technology and has been adopted in every virtually global market. From 1996 onwards, Europay International became part of MasterCard in 2002. Then JCB (formerly Japan Credit Bureau) joined EMVCo in 2004, and American Express in 2009. Since then, many banks not only have recognized the benefits of chip-based payment but also realized that helping foster global interoperability became necessary for the international standards. As a result, the EMV Specifications were created to fill that void.[3] It is more clear that EMV is an open-standard set of specifications for EMV chip payment cards and card accepting devices, such as point of sale (POS) terminals and ATMs. It can provide security and global interoperability by defining not only how smart bank cards can be used to prevent counterfeit fraud but also how they can be used for multiple payment applications on the identical cards.[1]

Nevertheless, the shift from magnetic strip to chip-based smart card payment has reduced the use of counterfeit cards domestically, the rising fraud abroad has more than compensated. The deployed systems inadequacies have thus affected many bank customers. According to the Association for Payment Clearing Services (APACS), the UK banks trade association, witnessed 169.8 million of fraud due to counterfeit cards in 2008, up 18 percent from that of the 2007 figure.[4] Unlike magnetic stripe transactions, where typically only the card's number and validity dates is tracked and processed. For each chip-based card transaction, it contains huge amount of piece of information to be exchanged between the card itself, the processing terminal and the acquiring bank's host. This requires the terminal to perform many stages of complex processing, ranging from simply reading application data to cryptographic authentication and verifivation, and to successfully complete an effective transaction, which means that adding support like developing a security model for EMV to the existing payment applications can be a challenge and daunting task.[5]

1.2 Aims and Objectives

Certainly EMV cards cost issuers much more than magnetic stripe, and eventually the acceptance infrastructure has to evolve into chip acceptance to derive the benefits of increased payment security, according to the study of US EMV business case.[6] Migrating the e-Banking market to EMV-based payment system is made more complex by the competitive, multi-tiered structure of the payments and acceptance industry. The shift applies to all bank transactions except paper-based ones such as cheques, and also covers credit card transactions. What's more, nowadays all the stakeholders are sharing higher costs driven by the growth of data breaches, online fraud and financial crime malware. Besides, with the soaring losses due to online banking fraud, the Chip Authentication Programme (CAP) is also introduced by the banks. Unlike EMV standard, the CAP standard is secret and was rolled out without any public scrutiny, despite being a critical security component the public must rely on for online banking transactions. Although it adds independent functionality for online transactions, the implementation of the CAP system is heavily based on the EMV smart cards protocol , which means that using EMV as the basis for CAP, not only reducing the development and deployment costs but making CAP devices that did not need to be personalised, according to Drimer et.al.[7]. For the reason of simplicity,this project will mainly focus on the analysis of EMV in general and regard CAP as a special case of EMV.

For the above analysis, EMV as an open-standard set of specifications for the smart card payments systems, has sparked wide interest amongst the cryptographers in recent years, which require the cryptographers to construct a more secure model for EMV, helping prevent the counterfeit card fraud and ensuring the security for the clients as well as the issuers and the merchants during the business transaction. Hence, the main aim of this project is to investigate the current advances in the area of EMV specifications and try to tackle the problem of why the current security model for EMV specifications are vulnerable to the attacks. Firstly, this project will examine the fundamental of EMV specifications and address the corresponding cryptographic functionalities that exist in the current literature in order to understand the efficiency and performance of EMV-based smart cards,and then it will illustrate what the potential attacks are for EMV-based payment systems and how the adversary can corrupt certain cards, finally it will develop a security model to enforce the EMV-based payment systems after introducing the general security model and security requirements. To summarize, the main objectives of this project are:

1 Reviewing the literature on the current infrastructure of EMV ;

2 Analyzing what the security property the EMV specifications have and realizing the potential attacks for EMV;

3 Discussing the corresponding cryptographic primitives involved in EMV specifications;

4 Abstracting the theoretical model for EMV based on the above cryptographic functionality and security model;

5 Researching the construction for the EMV specifications from the aspect of cryptographic primitives;

6 Analyzing and proving the obtained results.

1.3 Results

In general, this dissertation should be considered as a starting point to direct further research, which indicates that the project is more Type of III, namely theoretical and research one. For the project is mainly about developing a security model which aims to enhance the security of EMV-based transactions. In addition, this dissertation will concentrate on discussing and researching the current state of EMV, in order to understand what the central theories and the underlying cryptographic techniques involved in the EMV specications are, and then start to abstract the theoretical model to cover all the important options for EMV cards transaction and capture the adversary's game by defining the oracles for them. By analyzing and proving the involved cryptographic primitives for the construction, this dissertation exploits the security requirements and vulnerabilities regarding the design and implementation of EMV specifications. From the research result obtained, it appears that the security of the generic construction depends on the security of the involved cryptographic schemes. From this point of view, this dissertation can lay a well foundation for the future work to extend to include the issuer and make the pseudo-code executable for the real life EMV transactions.

1.4 Thesis Structure

Generally, this project will mainly focus on researching what the central theories are to define how the EMV specifications work and what the security model is, in order to develop a cryptographic security model for EMV. To narrow down its component issues, this dissertation is organized as follows. Section 2 introduces the fundamentals of EMV specifications and the cryptographic functionality of EMV; Section 3 examines the security requirements for EMV specifications and help understand the adversaries' power over the EMV transactions; Section 4 based on the knowledge of formal security model, a theoretical game-based security model is constructed to demonstrate what the security objectives the EMV specifications are in possession of and what kinds of games the adversaries can play in the EMV transactions; Section 5 researching the construction for the EMV specifications from the aspect of cryptographic primitives; Section 6 concludes this dissertation within the critical evaluation and the perspective for the future work.

2 An Overview of Theory

2.1 The Basis of EMV specifications

EMV is a common set of specifications publicly used for securing payments between EMV-based chip cards and the card readers, which can protect each identical card against security and interoperability threats globally. The EMV specifications demonstrate interoperability from two different aspects. On the one hand, it helps the EMV chip cards and the corresponding accepting devices defining their electro-magnetic and physical characteristics. In general, the chip-based EMV card is powered by the card accepting devices and requires the accepting devices to perform functions; On the other hand, it defines the related transaction data and protocol suite used for payments system. Initially, the EMV specifications were started by Europay, MasterCard and Visa in 1993. At the very beginning, only contact cards were standardized in the EMV specifications; nowadays, more contact-less card are gradually covered by the specifications. Currently the EMV specifications is controlled by EMVCo, which is a company owned by MasterCard, VISA, American Express and JCB. In addition, EMVCo is responsible for managing, maintaining and enhancing the EMV specifications, to ensure global interoperability of EMV-based chip payment cards integrated with the corresponding acceptance devices.[1, 8].(As shown in **Figure 1**) Even though EMVCo aims to maintain and extend the standards to help the payments system facilitate global interoperability and compatibility to up to date, the underlying security issues are mounting dramatically, such as counterfeit fraud of using EMV cards, especially for a real-time online banking system. In particular, the growing use of EMV for securing transactions has sparked wide interest and become hot topic among banking industry and cryptographic researchers, who attempt to enhance the security of transactions.

2.2 EMV fundamentals

Basically, EMV's primary purpose is to ensure global interoperablity and security for EMV-based payments system, which adds functionality over card authentication, cardholder verification and transaction authorization to improve security and compatibility of EMV cards. In addition, an EMV card is tamper-resistant for the reason of including a variety of capabilities upon both hardware and software, where an anticipation of tampering can be detected and reacted immediately. On the basis of EMV fundamentals, the study of EMV specifications becomes essential for the construction of a security model.

According to EMVCo, over 1 billion of EMV cards were in use in 2010. Typically, the migration from magnetic stripe cards to EMV cards has already been completed in the UK in comparison with other Europe countries. The reason why the transformation dramatically happened during the past decades is that the EMV-based transactions have significantly improved the security of the payments system. Basically, improvements are mainly achieved by adding the following basic functionality for the transaction cards, according to the research of *Smart Cards Appliance Council*[1]:

- Card Authentication (SDA, DDA and CDA) – the EMV specifications define three types of card authentication mechanisms: SDA, DDA and CDA, which can be used to protect

Figure 1: What is EMV?[8]

EMV cards secure against counterfeit and all mechanisms can be supported both by the EMV cards and the merchant terminal.

- Cardholder verification (signature and PIN)– the EMV specifications define five types of cardholder verification mechanisms: online PIN, offline plaintext PIN, offline encrypted PIN, handwritten signature, no cardholder verification. To simplify, the mechanisms can be categorised as online or offline PIN verification, and signatures;
- Transaction Authorization (online/offline)– the EMV specifications define two types of transaction authorization: online and offline. By using the pre-specified rules, the transaction can be performed offline by the card or online by the issuer.

According to Balfe, he concluded that EMV supports both Cardholder Authentication (SDA/DDA/CDA) through new Cardholder Verification Methods (CVMs) and ICC authentication [9]:

> ”*The introduction of new CVMs and SDA/DAA/CDA functionality aims to reduce credit and debit card fraud through improved authentication and authorization mechanisms. Additionally, depending on the results of card and terminal risk management routines, which are designed to protect issuers and acquirers respectively, transactions can be completed either online or offline. An online transaction is one in which the issuer is contacted to approve a transaction whilst an offline transaction is one that is completed locally between the card and the terminal based on a reconciliation of issuer and acquirer risk routines as stored in the card and terminal respectively.*”

2.2.1 Card Authentication

This project next consider how the EMV specifications support credit/debit cards authentication, which is generally defined as Card Authentication Methods (CAMs). The Merchant terminal can assure which bank has issued the current EMV card and whether the card specific data has been altered by performing card authentication. For each EMV card, authentication will be performed either online or offline, it is determined by the EMV card or the Merchant terminal when communicates with each other during the transaction. For the reason of simplicity, this project will emphasize the most used security mechanisms for the offline CAMs. Basically, there are two main types are supported by EMV: SDA and DDA, where CDA is considered as an enhance vision of DDA).

- **Static Data Authentication (SDA)** Basically, SDA provides protection for the integrity of EMV card's static data. According to Khu-Smith et.al [10, 11], a two-level key management hierarchy is supported by SDA. The top level of the hierarchy is Certification Authority (CA) which is known as the card scheme such as MasterCard and Visa, having its own pair of public-private key (P_{CA}, S_{CA}). As known, for the Issuer, it also has its own key pair (P_I, S_I). Hence, CA can use S_{CA} to sign a certificate ($CERT_I$) for the Issuer's P_I and store $CERT_I$ on the IC card. Moreover, the Issuer's SK_I can be used to sign the IC card's static data (SSAD) and also store on the IC card. For the Merchant terminal, it has in possession of the trusted copies of CA's public key (P_{CA}). By performing SDA, the EMV card sends the Issuer's $CERT_I$ and SSAD to the terminal, and the terminal then uses the obtained copy of P_{CA} to verify P_I. Furthermore, the terminal verifies the card specific static data pre-generated by the Issuer and stored in the IC card during personalization process. However, since SDA gives limited protection for the card specific data, it remains a possibility for cloning and replaying, which drives the card authentication methods (CAMs) shifting from SDA to DDA by having a terminal-generated nonce.

  $C \to T$: $(P_I, CA), P_I$ certified by CA
  $T \to C$: uses P_{CA} to retrieve the Issuer's P_I certified by the CA
  $C \to T$: (SSAD), static signed application data with Issuer's digital signature
  $T \to C$: Success/Fail, uses P_I to verify the SSAD to complete card authentication

- **Dynamic Data Authentication (DDA)** Compared with SDA, DDA provides dynamic protection for the integrity of EMV card's transaction data, and involves one more step than SDA. Similarly, DDA is also supported by a PKI as that in SDA, which Kuh-Smith et.al [10] defined a three-level key management hierarchy for DDA. Except for the top-level key pair for CA (P_{CA}, S_{CA}) and the Issuer's own key pair (P_I, S_I), DDA allows each EMV card has its own key pair (P_{IC}, S_{IC}) certified by the Issuer. For the top-level and second level CAs, they specify card schemes and the Issuer of the EMV card respectively. For the card-level, the private key P_{IC} is stored on each EMV card, and the Issuer's S_I is used to sign a certificate $CERT_{IC}$ for the card's public key P_{IC}, which is also stored on the EMV card. Hence, by performing DDA, the Merchant terminal first sends an INTERNAL AUTHENTICATE command (IAC) which contains an unpredictable number to the EMV card. Then the EMV card digitally signs the (IAC) data and sends it along with the Issuer's $CERT_I$ and the card's $CERT_{IC}$ to the Merchant terminal. When the Merchant terminal receives the data, it uses its stored copy of P_{CA} to retrieve the Issuer's P_I and then the P_I can be used to verify the $CERT_{IC}$ to obtain the card's P_{IC}. Finally, the EMV card's public key P_{IC} can then be used to verify the "dynamic" signature created by the card before. Within the terminal-generated nonce (unpredictable number), DDA can prevent EMV card from being cloned because of this challenge-response authentication. [12]

$T \rightarrow C$: INTERNAL AUTHENTICATE command (IAC) (contains an unpredictable number)
$C \rightarrow T$: $((P_I, CA), (P_{IC}, CERT_I, CERT_{IC}, IAC)$, P_I certified by CA and P_{IC} certified by Issuer
$T \rightarrow C$: uses P_{CA} to retrieve the Issuer's P_I certified by the CA
$T \rightarrow C$: uses P_I to retrieve the card's P_{IC} certified by the Issuer
$C \rightarrow T$: $Signature = Sign_{S_{IC}}(IAC)$, IAC dynamically signed by the card
$T \rightarrow C$: Success/Fail, uses P_{IC} to verify the signature to complete card authentication

Nevertheless, it still has drawbacks since it is not tied to the card authentication, the Merchant terminal cannot spot the fake transactions. As a result, Combined Data Authentication (CDA) is recommended to repair this by adding digital signatures over transaction data. Basically, CDA is essentially an enhanced vision of DDA, whose process is similar to DDA. While the main difference involves with the Application Cryptograms (AC), it is computed by using the EMV card as a transaction function and the results is then included in the transaction data signed by the EMV card as part of the card data authentication. [11] For this reason it can prevent certain DDA wedge attacks which will be discussed later, but it remains further research and seems nobody doing this yet.

2.2.2 Cardholder Verification

Generally, the Cardholder Verification Methods (CVMs) can be used to check the cardholder is the genuine one. As specified in the EMV specifications [13], for this step is optional, the Merchant terminal will first check the Application Interchange Profile (AIP) to determine if the EMV card is supported by certain CVMs. If the EMV card does support the Cardholder Verification, then the Merchant terminal process the Cardholder Verification Rules (CVR) as appeared in the CVM list. For each CVR, it determines which the associated CVM should be done based on the condition code of CVR to perform Cardholder Verification and states what has to be done if the Cardholder Verification cannot be completed. The Cardholder Verification can only be completed by the following circumstance[12]:

- CVM list is completed exhausted;
- One of the CVR did not completed successfully and no more CVR is available
- CVM is performed successfully.

After successful Cardholder Verification, the Merchant terminal will fill the Cardholder Verification Methods Field to indicate it before moving to the next step. Although there are five types of CVM are provided (*offline plaintext PIN, offline Encrypted PIN, online PIN, Signature, no CVM*), the CVM supported by the EMV card is offline PIN verification by the the card at

POS terminal, for online PIN verification is not standardized by EMV specifications. Moreover, since the EMV card is highly advertised as ”chip and PIN” cards, especially in the UK, this project will mainly describe the Cardholder verified by PIN as an example in the general payments model.

- `Offline PIN Verification` For offline PIN-based CVMs, the Cardholder enters a PIN and the PIN code is sent to the card chip application via a secure PIN pad, in which the Cardholder's PIN is verified and an unauthenticated PIN verification result is sent back to the terminal. Hence, the verification result indicates whether the entered PIN was correct or may further providing how many failed PIN attempts there are left before blocking current EMV card. Occasionally, the Merchant terminal may retrieve the value of the PIN Entry Counter (PTC) from the card chip application prior to VERIFY PIN command by issuing GET DATA command. [12]

  $T \rightarrow C$: GET DATA command (optional)
  $T \rightarrow C$: VERIFY PIN command
  $C \rightarrow T$: Success / (PIN failed, tries_left) / Failed

- `Offline Encrypted PIN Verification` For encryption the entered PIN from the secure PIN pad, the EMV card must have its own asymmetric key pair to support the Offline Encrypted PIN Verification, which can be the same pair as used for DDA. It also assumes that the RSA-capable EMV card has an RSA computation capability. The card first sends the terminal a certificate $CERT_{IC}$ for its public RSA key $P^r sa_{IC}$, the terminal can then obtain a copy of this $P^r sa_{IC}$ by verifying the $CERT_{IC}$ (or the PIN encipherment public key P_{PE}). Next, the terminal sends a GET_CHALLENGE command to the card and concatenates the PIN block obtained from the secure PIN pad, along with the response random number from the card (GET_CHALLENGE) and the unpredictable number generated from the terminal. Then this concatenated data is encrypted either with the $P^r sa_{IC}$ or P_{PE} by the terminal and sends to the card. [12] The card finally verifies this encrypted PIN by decrypting it with S_{IC} (or S_{PE}) and sends the PIN verification unauthenticated results back to the terminal as for offline plaintext PIN verification. Similarly, the Merchant terminal may retrieve the value of the PIN Entry Counter (PTC) from the card chip application prior to VERIFY PIN command by issuing GET DATA command.

  $C \rightarrow T$: $CERT_{IC}/CERT_{PE}$
  $T \rightarrow C$: $P^r sa_{IC}/P_{PE}$
  $T \rightarrow C$: GET_CHALLENGE command
  $C \rightarrow T$: (random number ($nonce_{IC}$), PIN)
  $T \rightarrow C$: GET DATA command (optional)
  $T \rightarrow C$: VERIFY PIN command ($PIN_{ENC} = Enc_{P_{IC}/P_{PE}}(pin, nonce_{IC}, nonce_T)$)
  $C \rightarrow T$: Success / (PIN failed, tries_left) / Failed

- `Online PIN Verification/Signature/No CVM` Typically, online PIN verification takes place at ATM and in order to support this verification method, ATM must map the PIN entry into a PIN block (64-bit) before being encrypted for transmission. Later, the PIN block under encryption will be decrypted by the acquirer and then encrypted again for the transfer via payments system. However, it is not covered in the newest EMV specification. Furthermore, for the terminal supports signature or no CVM of cardholder verification, this verification process is considered to be successfully completed.[11, 12]

2.2.3 Application Cryptograms

For EMV cards, the integrity and the original authentication of transaction data are provided by the Application Cryptograms (ACs) generated by the EMV card and i Message Authentication Codes (MACs) based on a shared secret key between the Issuer and the EMV card. Typically, ACs can protect transaction message generated by EMV card using AC short-term session keys, while a long-term MAC master key was shared between the EMV card and its issuer, known as IC card Application Cryptogram Master Key, MK_{AC}. Additionally, Application Cryptogram Session Key, SK_{AC}, is derived from MK_{AC} using a session key derivation function, in which the related keys will be discussed in the section of key infrastructure. Since the card's Application Transaction Counter (ATC) is used to keep track of the total number of performed transactions, the key SK_{AC} will be different for every AC generation.[9]

EMV specifications define four types of transaction messages depending on the completion of terminal risk management routines, then the following ACs will be generated either by the EMV card or the Issuer: Application Authentication Cryptogram (AAC), Authorization Request Cryptogram (ARQC), Authorization Response Cryptogram (ARPC) and Transaction Certificate (TC). During the transaction, the card generates one or two types of ACs, one AC (TC) for the offline transaction while two ACs (AAC/ARQC) for the online transaction. [8, 9] If the transaction is approved offline, then the card generates a TC and passes it to the Merchant terminal to claim payment. If the transaction is declined, the card generates an ACC. Moreover, if the transaction is approved online, then either an ACC or ARQC will be generated by the card. In both case, the transaction message (TC/AAC/ARQC) will be forwarded to the Issuer, and in turn the Issuer will reply with the message ARPC to indicate which method the current transaction should be used, then a TC or AAC will be generated by the card depends on the result of approval.

Basically, the Merchant terminal uses GENERATE_AC command to ask the card to compute one of the cryptograms based on the action analysis. The GENERATE_AC command specifies which one is the terminal's preliminary decision of the application cryptograms and whether CDA is involved. Formally, the cryptogram returned by the card is a MAC computed over the transaction details by using the shared symmetric key SK_{AC} between the card and the Issuer. Here, the transaction detail includes the transaction amount, terminal country code, terminal verification results, currency, date, transaction type, terminal-generated nonce provided by the terminal, plus AIP and ATC provided by the card. [8] Moreover, the MAC technique used in EMV specifications is a CBC-MAC computed based on DES in CBC mode with triple-DES applied to the last block, according to [13], and the cryptographic technique for EMV will be further discussed later. Hence, it can be concludes as below:

$T \rightarrow C : GENERATE_AC$(TC, cda_requested, specified data)
$C \rightarrow T$: TC = (CID (Cryptographic Information Data), ATC,

MAC, IAD (Issuer ApplicationData, optional))

The Application Cryptograms start with GENERATE_AC command sending from the Merchant terminal to the card to indicate what type of AC is requested and whether a CDA signature is requested, in which the $MAC = MAC_{SK_{IC}}$(the transaction amount, terminal country code, terminal verification results, currency, date, transaction type, terminal-generated nonce, AIP and ATC).

So far, this section provides a brief overview of the current EMV specifications, particularly in relation to EMV Key Infrastructure and Protocol Phases, which lays a well fundamental for the further research. In addition, the details of the technical introduction for EMV specifications were included in the newest EMV specifications (EMV 4.2 published in 2008), which contains *Application Independent ICC to Terminal Interfave Requirements*[14], *Security and Key Agreement*[13], *Application Specification*[15], *Cardholder, Attendent, and Acquirer Interface*[16]. These four books define the EMV specification as a protocol-suite for payments system instead of a single protocol.[8] By studying with Book 2 *(Security and Key Management)* of EMV 4.2[13], it helps understand the key infrastructure, card authentication mechanisms (CAMs) along with data object lists(ODL) among other standards. Besides, for Book 3 *(Application Specification)*[15] based on the functionality outside the application layer described in the Book 1 *(Application Independent ICC to Terminal Interface Requirements)*[14], it mainly concentrates on the functions describing what has happened after the initial application selection. Concurrently, this project will mainly focus on Book 2 & 3 to design our construction.

2.3 Cryptographic functionality for EMV protocols

Basically, the underlying security framework for EMV specifications is mainly based on two cryptographic techniques (symmetric and asymmetric techniques), which can protect EMV cards secure against fraud for both offline and online transactions. First of all, the symmetric authentication keys help leverage the security for chip-based cards and the access to the card's memory. On the other hand, asymmetric keys and certificates are incorporated to facilitate card authentication in offline transaction environments.[17] By understanding how EMV cards work for protecting against fraud and what cryptographic functionality EMV can provide, it is essential to introduce the basic building blocks of EMV and some central concepts and protocol session.

2.3.1 Key Setup

Basically, the general model consists of a customer and a merchant who exchanges money for staffs through the payments system. In addition, the customers and merchants communicate over an open network with each other and also with their issuing bank and acquiring bank.[18], which can be shown in the figure 2.

During a transaction, actual connectivity may include but not limit to the customers and the merchant. In a typical purchase scenario, it seems that the customers only has a connection to the merchant, however, he also indirectly connected communicated with his issuing bank through forwarding an authorization message to them.

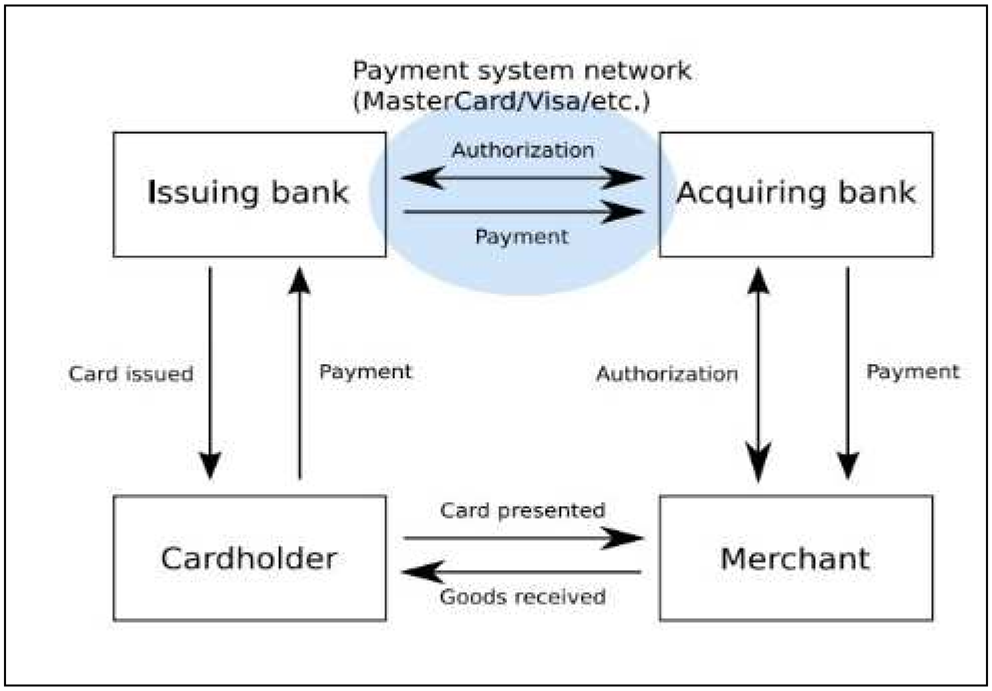

Figure 2: Smart Cards Payments System[20]

Certainly, the number and type of keys for EMV-compliant cards (ICC) depends on the authentication methods supported by the EMV card. For the EMV card supports SDA, it maintains the issuer's P_I signed by the card trusted CA, along with a signature to store SSAD which is created by the issuer's S_I. In addition, the terminal stores P_{CA} (the public key of CA) in order so that it can be used to verify SSAD.[9] Figure 3 shows the relationship between cryptographic keys and certificates for SDA. where the issuer personalizes its SDA cards with the IPK Certificate received from the CA and a signature of the static data from the card.[19]

While the EMV card supports DDA, it has its own key pair whose public component is stored in $Cert_{IC}$ and private component is also used to sign dynamic authentication data (SDAD). Similarly, the EMV card for DDA maintains the issuer's P_I and store SDAD signed by the issuer's S_I. Figure 4 shows the relationship between cryptographic keys and certificates for DDA. where the issuer personalizes its cards with the IPK Certificate received from CA, the ICC Public Key Certificate, and the secret ICC Private Key which is stored in a secure confidential location on the card.[19]

Similarly, the terminal for DDA also uses P_{CA} (the public key for CA) to verify the issuer's $Cert_I$ and then uses the verified P_I to verify the EMV card's $Cert_{IC}$ which stores P_{IC}. Moreover, in the case of DDA, the EMV card's private component S_{IC} can be used to enable challenge-response procedure, which can also be used to sign the application data dynamically and generate a random number provided by the terminal. Then, the terminal determines whether the card is the genuine one or not as well as to ascertain if the data personalized in the EMV card has been altered since card personalization.[9]

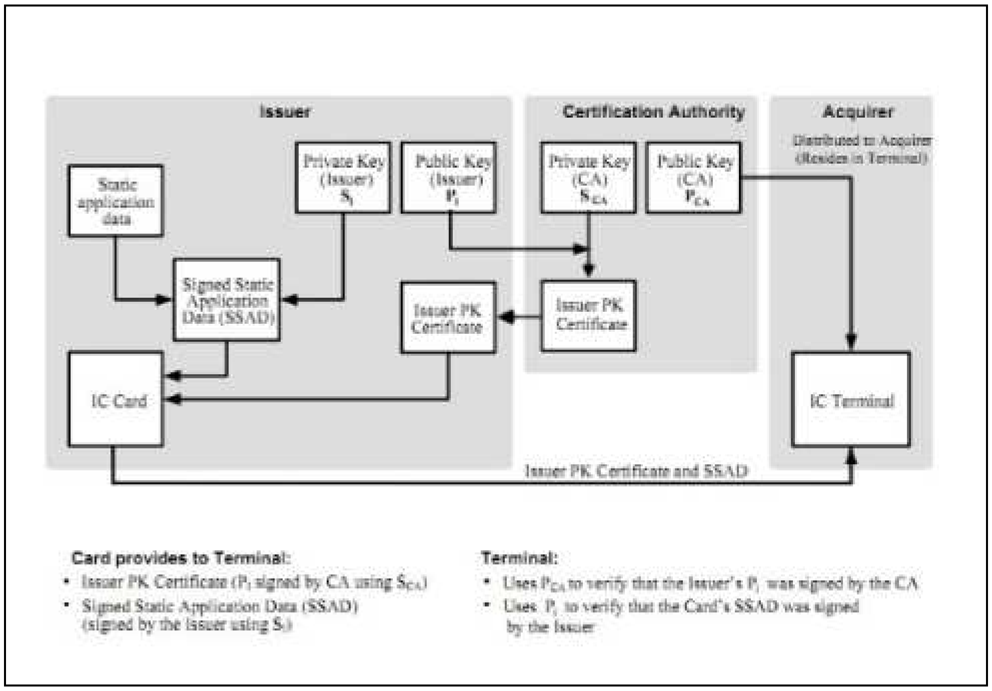
Issuer
Certification Authority
Acquirer
Distributed to Acquirer (Resides in Terminal)
Static application data
Private Key (Issuer) S_I
Public Key (Issuer) P_I
Private Key (CA) S_{CA}
Public Key (CA) P_{CA}
Signed Static Application Data (SSAD)
Issuer PK Certificate
Issuer PK Certificate
IC Card
IC Terminal
Issuer PK Certificate and SSAD
Card provides to Terminal:
• Issuer PK Certificate (P_I signed by CA using S_{CA})
• Signed Static Application Data (SSAD) (signed by the Issuer using S_I)
Terminal:
• Uses P_{CA} to verify that the Issuer's P_I was signed by the CA
• Uses P_I to verify that the Card's SSAD was signed by the Issuer

Figure 3: Diagram of Offline Static Data Authentication (SDA)[13]

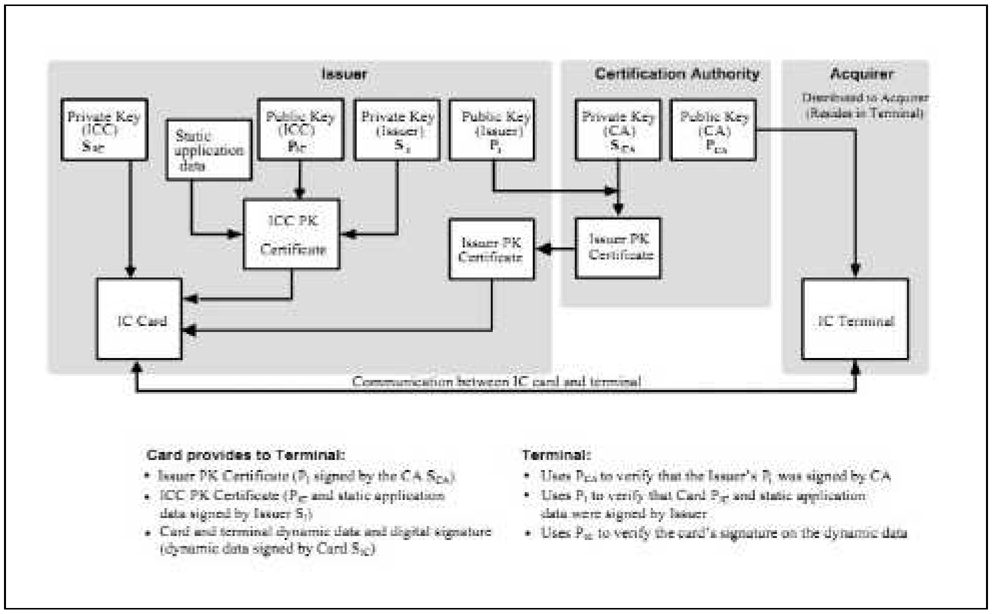
Issuer
Certification Authority
Acquirer
Distributed to Acquirer (Resides in Terminal)
Private Key (ICC) S_{IC}
Static application data
Public Key (ICC) P_{IC}
Private Key (Issuer) S_I
Public Key (Issuer) P_I
Private Key (CA) S_{CA}
Public Key (CA) P_{CA}
ICC PK Certificate
Issuer PK Certificate
Issuer PK Certificate
IC Card
IC Terminal
Communication between IC card and terminal
Card provides to Terminal:
• Issuer PK Certificate (P_I signed by the CA S_{CA})
• ICC PK Certificate (P_{IC} and static application data signed by Issuer S_I)
• Card and terminal dynamic data and digital signature (dynamic data signed by Card S_{IC})
Terminal:
• Uses P_{CA} to verify that the Issuer's P_I was signed by CA
• Uses P_I to verify that Card P_{IC} and static application data were signed by Issuer
• Uses P_{IC} to verify the card's signature on the dynamic data

Figure 4: Diagram of Offline Dynamic Data Authentication (DDA)[13]

Hence, according to Ruiter, J.D. et.al research paper and Book2 *(Security and Key Management)* of EMV 4.2, the general Key Set-Up for EMV specifications between different parties can be concluded as following based on the above discussion:

– For each EMV card, a key derivation function generates an unique symmetric key MK_{AC} under the issuer's master key MK_I. In general, MK_{AC} is shared between the EMV card and the issuer. And then a session key SK_{AC} is computed based on the generation of MK_{AC}.

– For the issuer, it has in possession of a public-private key pair (P_I, S_I),and the terminal who is considered as the trusted media knows this public key component P_I.

– Similarly, for EMV cards which support asymmetric crypto, they also have a public-private key pair (P_{IC}, S_{IC}) used to sign the authentication data.

More general, EMV offers both asymmetric and symmetric key mechanisms: on the one hand, asymmetric key mechanisms authenticate the EMV-based smart card as a valid card to the terminal; on the other hand, symmetric key mechanisms generate and verify transaction cryptograms based on the key shared between the customer and the issuer, essentially for Message Authentication Codes(MACs). [18] Certainly, the key setup is the basis of trust between the involved parties.

As is known, the infrastructure of both symmetric and asymmetric key mechanisms can help EMV cards prove the authenticity of transaction data , which can be concluded from two aspects. On the one hand, because of the sharing of symmetric key MK_{AC} between the issuers and the EMV cards, all EMV cards are allowed to put MACs(Message Authentication Codes) on messages, where the MACs are used to verify the authentication of messages on behalf of the issuers; On the other hand, for cards that support asymmetric crypto,it allows the customers' authenticity can be proved through their digital signature. The difference between the two key mechanisms is not only about if the EMV card would either support the symmetric key or the asymmetric key, but it is also about the choice of the terminal that one of which mechanisms would be used to authenticate the transaction messages. After showing the details of the underlying key infrastructure for the involved entities, this dissertation will then introduce a number of protocol phase to identify how EMV specification works for the banking transactions.

2.3.2 Protocol Phases

The processing of EMV-based transactions starts when the card is inserted into an accepting devices (e.g. POS terminal), which then involves in a number of protocol phases. Generally, an EMV protocol session can be roughly divided into the following steps:

1 *Initial Decision:* is generally known as reading application data. By reading application data, the terminal obtains the knowledge of the EMV cards' configurations, moreover, the terminal also provides the card with related transaction data optionally.

Initially, the protocol begins with SELECT_APPLICATION command which is completed by the EMV cards, returning a Processing Options Data Object List (PDOL) to indicate the selection results to the terminal. Then, a READ_RECORD command returns the terminal with basic information about the card itself, like the cardholder's account details, and information about the applications supported by the card and their configurations, such as the issuer and the card's public key and certificate or some signatures data under the issuer's key (SDA only).[8] Finally, the terminal then uses these information to build a list of possible payment types of cardholder verification methods (CVM list), which can be used for the cardholder verification later. If the EMV card and the terminal both support more than one payment types, the cardholder may select the application to be used from the previous CVM list.[8, 12] Note that the information obtained in this stage has not been authenticated by any means. However,these information is necessary for the following steps in the EMV session.

2 *(Optionally)Card Authentication Mechanisms:* by means of SDA,DDA, and CDA to check that the card is genuine.

A main advantage of using EMV cards for payment systems is the reduction of the losses caused by counterfeit cards. As a matter of fact,this is achieved by introducing more effective Card Authentication Methods (CAMs).[12] After the initial session of protocol phase, the terminal learns general information about which authentication mechanisms are supported by the EMV cards. In addition, the terminal also makes choice about the related cardholder verifications methods (CVMs) agreed by the EMV cards in the subsequent steps of protocol phase.[12, 20] As already mentioned, the purpose of CAM is to prove that the card is genuine. In general, there are two types of card authentication mechanisms (CAMs): online and offline. As the online authentication mechanism is not standardized in the EMV specifications, this report will explain the most options and variants of the security mechanisms in the offline CAMs for EMV-based transactions.

Basically, Offline CAMs makes good use of public key cryptography. For example, the card issuer, and the EMV card both have a public-private key pair, known as (P_I, S_I) and P_{IC}, S_{IC}, which can be used to encrypt information that needs to be kept secret from the first stage, and also can be used to Offline Encrypted PIN verification process later. Here, a RSA-based digital signature scheme is involved in two ways: firstly they allow an entity to prove that integrity of data to the counterpart, which means that the data has not been altered during the transaction, and secondly to prove the confidentiality of related transaction data, which implies that the data really came from the legitimate entities. Moreover,as mentioned before, the concept of digital signatures also allows Public Key Infrastructure (PKI) schemes to be used, in which both a two-level hierarchy and a third-level of hierarchy based on CA (Certification Authority) is provided for SDA and DDA, respectively.[12] For the details of the defined authentication mechanisms:SDA, DDA and CDA, it can be found in the previous section (section 2.2.1).

3 *(Optionally)Cardholder Verification:* by means of PIN or signature. One of the main motivations of using EMV-based chip cards for credit and debit payment systems is the ability to support cardholder verification methanisms (CVMs) to authenticates the cardholder and protects against the use of a lost or stolen card. The EMV card provides the terminal with a list of CVM Rules, which specifies what CVMs are acceptable under certain conditions. The terminal then chooses the CVM to be used. Typically, Cardholder verification can be done in two ways: by a PIN or a signature.

- online PIN, the cardholder enters a PIN via a secure PIN pad from the trusted terminal and then the issuer checks the correctness of PIN entry for the involved EMV card.
- offline(plaintext) PIN, namely unencrypted PIN, the PIN entered via a secure PIN pad from the trusted terminal is transmitted to the card chip in plaintext format,

which can be validated under the chip.

– offline (encrypted) PIN, the cardholder's PIN entry is encrypted by using public key mechanism. Then the encrypted PIN is transmitted to the chip for validation, only if the card knows the private key can validate this PIN entry.
– signature, a written signature is required in a paper receipt. The merchant compares the signature on the receipt with the signature on the back of the card.

According to [1], the offline PIN is stored securely on the card instead of sending to the issuer host, while the online PIN is being sent online for the issuer to validate without storing on the card. Besides, the further difference for offline and online PIN verification is that the card is only involved in cardholder verification in case of offline PIN (plaintext or encrypted), while the card is not involve in the case of online PIN. Furthermore, offline PIN is the only method of cardholder verification supported by EMV that is not available with magnetic stripe cards.[1] Finally, the cardholder verification is completed by reporting success or failure through the card.

4 *Transaction Authorization:* by means of TC (Transaction Certificate), ARQC (Authorization Request Cryptogram) and AAC (Application Authentication Cryptogram). In the final session of an EMV protocol phase, after successful card authentication and cardholder verification, the actual transaction is performed through generating unique Application Cryptograms based on the results of approval or decline for processing the EMV transactions by the card.[12]

Transaction Certificate(TC)–it is generated by the EMV card when a transaction is approved, and proves that the transaction was authorized, either online by the issuer, or offline by the EMV card. Additionally,it helps protect the issuer secure against cardholder repudiation.

Authorization Request Cryptogram(ARQC)–it is sent to the issuer during an online authorization request, and is used to prove to the issuer that the card being used in the transaction is genuine. It is the cryptogram used by the issuer to perform online CAMs.

Application Authentication Cryptogram(AAC)–it is generated by the EMV card when a transaction is declined, and proves that the transaction was declined, either online by the issuer, or offline by the chip card or the terminal.

EMV transactions can be authorized either online or offline, the card and terminal communicate to determine whether the transaction can be authorized based on some certain rules. In both online and offline transactions, the card can also choose to refuse or quit the transaction, in which case an AAC is provided instead of TC or ARQC.

2.4 General payments model

Based on the above brief introduction of EMV specifications, this section will emphasize EMV-based general payments model to have an insight into EMV-based IC card transaction flow. Basically, an EMV payment transaction involves interactions between the following parties: the Cardholder, the Merchant, the Issuer and the Acquirer.[21]

Cardholder: An authorized holder of an IC card supplied by the Issuer. The card is in possession of the holder's payment details and generating authentication data and verifying the holder's PIN. Generally, the holder is capable of using a Primary Account Number (PAN) stored on the IC card to identify the holder's account information and authorize the Issuer. During a transaction process, the Cardholder only connected to the Merchant to pass the authorization messages to the Issuer via the Acquirer.

Merchant: This builds a relationship between the Cardholder and the Merchant and allows the Cardholder to purchase goods. Typically, it uses a terminal to connect with the card and the terminal also interacts with the Issuer via the Acquirer to receive the authorization messages to enable a valid transaction.

Issuer: As mentioned above, it is a financial institution that issues a IC card to the legitimate holder.

Acquirer: It is also a financial institution that performs transaction authorizations for the involved parties and processes IC card payments for the Merchant. Usually it supposes that the Issuer and the Acquirer communicate through a trusted secure channel.

The processing of an EMV transaction begins when the Cardholder inserted his IC card into the Merchant terminal and then the terminal read necessary application data from the card through a number of READ RECORD commands.

Step 1 & 2 Generally, these application data will be used in IC card's risk management procedures and established card authenticity. The transaction flow is shown in figure... In the transaction description, $X||Y$ denotes the concatenation of X and Y.

Step 3: In order to authenticate the IC card's public key in $Cert_{IC}$ (essentially performed via SDA), the Merchant first verifies the Issuer's public key certificate $Cert_I$ by using its copy of the CA public key. Hense, the Issuer's signature on $Cert_{IC}$ can be verified through the Issuer's public key obtained from $Cert_I$. [21] After successful SDA, the Merchant terminal (server) generates a purchase random number (PRN) and sends a challenge to the card using INTERNAL AUTHENTICATE command (IAC), then constructs a Purchase Information which contains a description of goods along with the purchase details such as the price for goods and time-stamp. Then the Merchant sends $IAC||PI$ to the Cardholder system.

Step 4: Upon receipt of the message in Step 3, the Cardholder system displays the purchase information to the Cardholder to confirm the purchase and forwards IAC to the card to check the presence of EMV card.

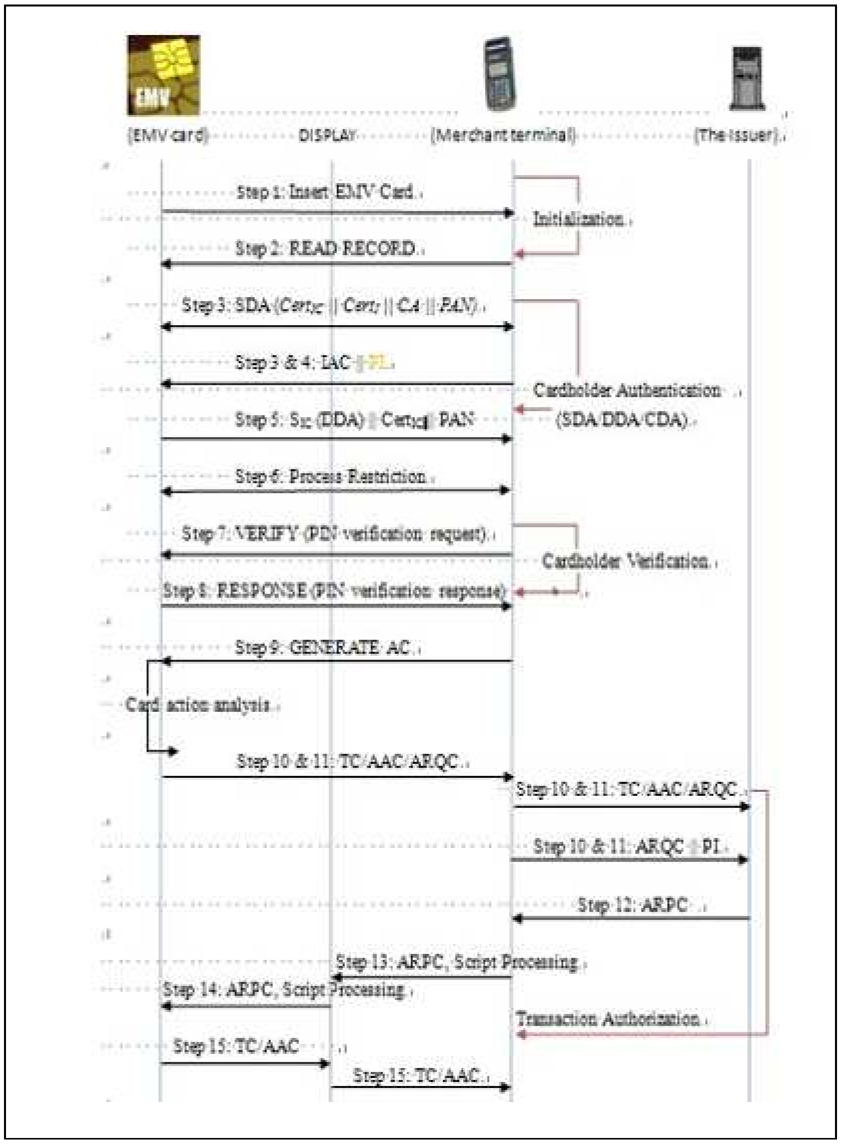
EMV
(EMV card)
DISPLAY
(Merchant terminal)
(The Issuer)
Step 1: Insert EMV Card
Initialization
Step 2: READ RECORD
Step 3: SDA (Cert_IC || Cert_I || CA || PAN)
Step 3 & 4: IAC || PL
Cardholder Authentication (SDA/DDA/CDA)
Step 5: S_IC (DDA) || Cert_IC || PAN
Step 6: Process Restriction
Step 7: VERIFY (PIN verification request)
Cardholder Verification
Step 8: RESPONSE (PIN verification response)
Step 9: GENERATE AC
Card action analysis
Step 10 & 11: TC/AAC/ARQC
Step 10 & 11: TC/AAC/ARQC
Step 10 & 11: ARQC || PI
Step 12: ARPC
Step 13: ARPC, Script Processing
Step 14: ARPC, Script Processing
Transaction Authorization
Step 15: TC/AAC
Step 15: TC/AAC

Figure 5: General Payments Model (Source: Fu, September, 2011, University of Bristol)

Step 5: After receiving IAC, the EMV card computes the signature of $S_{IC}(DAD)$ and sents it to the Merchant terminal. Here, DDA represents Dynamic Data Authentication including the random number (PRN) generated in Step 3. The Merchant terminal uses the card public key obtained from $Cert_{IC}$ in Step 3 to verify the signature $S_{IC}(DAD)$.

Step 6: When successful verification of signature has received, it implies that EMV card authentication has been completed successfully. Then Process Restriction[10] is perform, acting as a primer for the action analysis that aims to decide whether the transaction should be continued online or offline. Additionally, this mandatory step is performed by the Merchant terminal and does not need direct interaction with the EMV card.

Step 7: After successful EMV card authentication, the Cardholder verification method is invoked to ensure that the person who presents the EMV card is the genuine one. For this purpose, the Merchant terminal then requests the Cardholder to enter the PIN, which can be verified either online by the Issuer or offline by the EMV card itself. If the PIN verification is performed by offline, then the Merchant terminal sends a VERIFY request command to the EMV card.

Step 8: Upon the receipt of the PIN verification request, the EMV card returns a PIN verification response message (RESPONSE), which indicates success or failure of the PIN verification process. However, if the PIN verification is performed by online, then it shall be protected upon entry by encipherment according to ISO 9564-1[22], and the Merchant terminal will transmit the encipherment which contains the holder's PIN to the Issuer based on the payment rules.

Step 9: Until now, the Cardholder Verification is successfully completed. Next, the Merchant terminal passes a GENERATE AC request command to the EMV card to invoke the its action analysis depending on the EMV card Risk Management. The EMV card Risk Management aims to protect against fraud or excessive credit risk. Typically, there are three results can be returned from the action analysis: *approved offline, rejected online and processed online*.

Step 10,11: If the EMV transaction is *approved offline*, the EMV card generates TC and sents it along with the PI to the Merchant terminal and then forwards to the Issuer. The Issuer shall be verify the MAC in the TC by comparing the information in the TC with that in the PI to determine the processing of the transaction. If the EMV transaction is *declined offline*, the EMV card generates ACC and sents it to the Merchant server. Then the transaction ends. If, on the other hand, the EMV card decides to proceed the transaction online (*processed online*), it generates an ARQC and sends it along with PI to the Merchant terminal, and then forwards to the Issuer.

Step 12: The Issuer responds the ARQC with an ARPC to indicate the transaction will be accepted or declined.

Step 13,14: During the transmission, the Script Processing may also be performed by the Issuer, in order to send command scripts to the EMV card[10], allowing updates to the EMV card in the field. As a consequence, the Issuer can return the scripts via the Merchant terminal to IC card.

Step 15: If the transaction is accepted, the EMV card generates a TC and perform the previous process under approved offline transaction. The TC/AAC is then forwarded to the Merchant server to end the transaction.

3 General Security Objectives

Basically, banking payment system should provide high level of security as they manipulate some sensitive information both for the customers and financial institutions themselves. In general, they must ensure that the sensitive information relates to the customers and the transactions can not be exposed to the third parties, which means that smart card is only accessed by the legitimate entity and the bank is able to trace and verify where the transactions took place. In order to secure a session between the customers and the bank, it is necessary to authenticate them both. Consequently, the security objectives for EMV specifications have become significant. Hence, this section will describe the security objectives from two main aspects: on the one hand, it will generalize the security objectives of EMV transaction and discuss the adversary's power; on the other hand, it will outline a general introduction of formal cryptographic security model. Before formulating security requirements for EMV-based transactions, the section will initially make a number of assumptions between the involved parties to act as a primer.

3.1 Assumptions for Involved Parties

A1. The issuers and the acquirers enjoy and share a mutual degree of trust and infrastructure for securing transaction.
A2. The acquirer obtains a certificate from CA's public key and this certificate is loaded into the merchant terminal supported by this acquirer.
A3. The contract determines the business, responsibility and liability relationships between the merchant and the acquirer as well as the cardholder and the issuer.
A4. The cardholder (customer) can trust the payments system to enable a valid transaction.
A5. Every issuer and merchant terminal are all equipped with a copy of CA's public key in order to verify acquirer's certificates.
A6. Every merchant terminal is also equipped with its own signature key pair, and is provided with a certificate for its public signature verification key, signed by the acquirer.
A7. In a standard EMV transaction, terminal risk management is required to help protect the payment system against fraud, which can provide issuer authorization and ensure transaction security.

3.2 EMV Transaction Security

Basically, all participants (the customers, the issuer, the acquirer and the merchant) involved in any transactions would prefer not to suffer any loss so that it is necessary to define the general security requirements precisely for EMV specifications before designing the cryptographic security model. Typically, the security requirements can be divided as following[23]: authentication, confidentiality, integrity and non-repudiation respectively.

– **Authentication** Generally, according to Al-Meaither, et.al.[23], an entity authentication provides assurance to one party with respect to the involved and actually participated second party. Basically, the card authentication to the merchant terminal is performed by SDA/DDA/CDA, which is based on the knowledge of digital signatures. In standard EMV transaction, a top-level certification authority (CA) is established from the card scheme, which is known as the chain of the trust for EMV (which will also be discussed in the PKI). As mentioned in the assumption (A5), it is assumed that each merchant terminal has obtained a trust copy of CA's public key. In addition, the CA can sign the issuer's certificate for the public key which is stored on the EMV card. As a result, the CA's public key and the issuer's certificate can be used to verify the authenticity of transaction data stored and sent by the card. [10, 23]

And the cardholder authentication is achieved using a PIN entry obtained from the terminal. As is known, the PIN can be verified both offline and online, by the card and the issuer respectively. If offline PIN encryption verification is supported, then the merchant terminal performs an asymmetric encipherment mechanism, in which the card itself may have a separate RSA key pair or the signature key pair for PIN encryption/decryption. However, the merchant terminal authentication is not specified in the protocol. Then the above cardholder and EMV card authentication are provided to help issuer verify the merchant terminal and transaction message along with the legitimacy of the payment card to the merchant terminal.

– **Confidentiality** The aim of confidentiality is to prevent unauthorized entities to gain the knowledge of transaction related data, which requires that the cardholder should keep his PIN secretly as well as his privacy order from unauthorized entities. Furthermore, it is also about keeping the asymmetric key (the private component) and the shared symmetric key for the involved entities during the transaction process.

– **Integrity** The integrity in anticipation of ensuring transaction data can be protected from an unauthorized entity, which also implies that the cardholder's payment authorization must be protected against alteration.

  Since the EMV support offline encrypted PIN verification, the confidentiality and integrity of entered PIN can be protected. Although, the entered PIN is not encrypted, its confidentiality and integrity can also be ensured by the cardholder, which has made the environment without over control. The Application Cryptogram (AC) integrity is guaranteed by using MACs, which are computed by using a session key (SK) derived from a long term secret key shared between the card and the issuer (MK_{IC}, IC card's master key), while the AC confidentiality is not specified in the EMV standards.[10]

– **Non-Repudiation** Basically, it helps prevent an entity from denying the purchased transactions. For instance, a customer may claim that a recorded transaction is not made by him, then the bank should be able to prove if the customer's statements is true or false.

  Basically, the cardholder's non-repudiation is provided by the TC, which can provide evidence if the cardholder authentication process has taken place or not. Hense, it allows the cardholder to enter his/her PIN under the consent of this valid and legitimate transaction. On the contrary, the merchant terminal's non-repudiation is not provided by the EMV specifications.

3.3 Attacks on EMV protocols

As the discussion of EMV specifications, it enhance the understanding of what smart cards based on EMV enable to do during a financial transaction, and how the cryptographic functionality works to ensure the integrity and confidentiality of transaction information. Basically, EMV specifications for smart cards have been gradually accepted and adopted since it was firstly introduced in the 1990s, which have been fully deployed in the UK since 2006 the security of EMV-based transactions have been widely advertised. However, the soaring development of EMV-based payment system has also aroused the researchers to observe the weakness and vulnerabilities for EMV standards. Thus, several security flaws have been identified by researchers and remained more future research to improve the security of EMV-based payment system. In this section, it will describe the attacks on EMV protocol-suite to capture the adversary's capabilities. From the previous research of cryptographic functionality embeded in EMV protocol, the possible attacks on EMV can be roughly discussed as following.

3.3.1 Evasdropping Attack

For one of the advantages for a EMV-based smart card is that it is very convenient to carry, then it may lack a trusted application interface, which cannot provide the cardholder's PIN to the card directly. Thus, the EMV card is vulnerable to eavesdropping attack.[24] For example, if the arbitrary cardholder's account and the corresponding PIN entry are eavesdropped during an EMV transaction at the POS terminal, then an adversary can fraud these transaction data into his or her magnetic stripe card, and then use it in a foreign country where the previous EMV card is not presented. That is to say, a fake can be created for a stolen or lost card, in which offine transactions is accepted by the POS terminal, with any PIN entry. Moreover, Bond[25] concluded that there are multiple approaches to eavesdropping POS termianl, for instance, a camera and double-swipe, the camera is positioned with the view of PIN pad, and secretly swipes the card through his own equipment before inserting it into the genuine device.

3.3.2 SDA Card Cloning

As is known, EMV heavily depends on the use of PIN in order to authorize transactions, according to Murdoch et.al.[26], SDA Cards are vulnerable to a trivial and well-known replay attack. For the lack of freshness mechanism in SDA, the certificate for PIN verification can be read from one legitimate card and then written to another counterfeit card, which increases the possibility of sensitive data being copied between EMV cards. More generally, this is known as ”yes” cards, which indicates that no matter what PIN is given by the adversary, the certificate always responses ”yes” for the PIN verification request. Moreover, since SDA card is not able to perform RSA public key mechanism, the certificate for PIN verification request can only be signed statically. By signing certificate without dynamical property, MACs used for provide data integrity and confidentiality cannot be checked by the terminal at a real-time network. Hence, it would be better that the EMV cards should support DDA/CDA instead of SDA, to help sign the certificate dynamically by using the RSA public-private key pair to discover the clone cards by online transaction authorization.[20] However, the transformation from SDA to DDA/CDA also increases the communication costs and becomes a burden for the merchants, although ”yes” cards can be detected normally under the use of DDA/CDA cards.

3.3.3 Wedge Attack

It is clear that the introduction of DDA for cardholder authentication is a big improvement, but it's not perfect enough. As a matter of fact, EMV card authentication occurs before PIN verification, then it can be exploited with a "wedge", which implies that EMV cards are also vulnerable to wedge attack. Generally, the issuers is not capable to detect that a wedge attack has occurred during the online authorization process, which implies that wedge attack is possible both for online and offline transactions for SDA and DDA cards as well.(cited in [27]) In this way, the adversary who obtained the lost or stolen card can use it processing the transactions without knowing an correct PIN. However, it is not the same as "yes" cards attack, which attempts to accept any PIN entry, which works only for which the terminal dose not contact with the acquirer before completing the transaction. On the contrary, for wedge attack, it may happen both online or offline transactions. The existence of wedge attack requires the banking industry to shift from DDA to another variant of card authentication–CDA, which is believed to solve this problem partially. However, CDA is too new to make good use of it, in which wedge attack needs to be improved in the future research.

3.3.4 Man-In-The-Middle (Replay) Attack

Furthermore, except for the adversary obtains the cardholder's PIN, the adversary can also corrupt the terminal, for instance, the adversary can modify the terminal's configurations. Then the modified terminal is capable of obtaining the copy of the cardholder's PIN as well during the legitimate transaction, which is known as man-in-the-midddle attack or replay attack. According to Markantonakisa et.al.[28], the attack occurs when a legitimate card is processing its financial transaction in a modified terminal so that the fraud can forward his/her transaction to the legitimate card without the notice of the cardholder. Then the fraud's transaction can be completed through terminal can not only perform on the current transaction information but can also communicate with a card emulator through such as Ethernet link. Typically, the emulator is positioned on another legitimate terminal within a the emulator and the modified terminal, in which the cardholder believes that he just purchased his own transaction. As a matter of fact, he had no idea that he has suffered one more authorized transaction which did not belong to him, which means that there is no any trusted way to verify whether the terminal has been tampered with and to show what transaction has been authorized on the cardholder's card. In summary, even the countermeasures against this attack exists, such as the use of tamper resistant terminals, "man-in-the-midddle" or "replay" attack is still hard to prevent.

Nowadays, EMV cards have gradually taken the place of magnetic strip cards for point-of-sale(POS) and ATM transactions among most European countries, and America and Canada are also gradually shift from traditional magnetic strip cards to EMV-based smart cards. Nevertheless, the migration requires that the soring use of EMV cards have to face the challenge of the attacks both for online and offline transactions. Basically, the banking industry and the researchers have claimed that most of these attacks were already known. On the basis of the published articles from Computer Laboratory at University of Cambridge, Murdoch et.al.[26], they suggested that smart cards based on EMV have just simply reduced cards fraud, while they have not eliminated it.

3.4 Formal Security Model

The advent of information technique has brought in many new opportunities for the booming of financial industry. The most significant one is the adoption of EMV specifications for banking transactions, which has almost taken the place of traditional magnetic stripe cards to secure card transactions and been widely used in most Europe countries. However, the growing use of EMV-based smart cards have identified and proved that there are several flaws and vulnerablities for EMV standards, which opens new research field of information security. The aim of EMV research is to protect the customer's transaction information secretly from arbitary attackers, while still allows legitimate customers to perform transaction with the issuer or the merchant freely. According to Murdoch et.al.[26], the EMV specifications authenticate both the card and the customer to secure card transactions, where transactions details are authenticated through a combination of cryptographic meaasge authentication codes(MAC), digital signatures, and the entry of a PIN. Thus, cryptographic primitives are involed in the research of enhancing EMV standards, which is known as a branch of information security and covers the study of algorithms and protocols that secure data. This section provides the definition of formal security model generally, which act as a primer for the generic construction for theoretical model for EMV.

Basically, on the basis of studying [29], a formal security model should take the following two aspects into account: firstly, it must specify how an arbitrary and probabilistic attacker can interact with legitimate users of a cryptosystem in the polynomial-time. In general, this stage involves in using the random oracle model, like hash function, to model certain parts of a given cryptography system; Secondly, it must state what that attacker should achieve in order to 'break' the cryptography system. Based on the two aspects, there are two approaches can be used to define the formal security model, namely, game-based model and simulated model.

3.4.1 Game-Based Security Model

According to Dent[29], for the game-based security model, the attacker initially interacts with a hypothetical probabilistic algorithm, which is known as challenger who generates all the corresponding secret keys used in the cryptosystem. In general, the challenger may respond to the attacker's queries later. Then, the cryptosystem terminates when the attacker terminates, and finally indicates whether the attacker has broken the cryptosystem by querying certain random oracles. Several game-based security models have been proposed and used in practice, the most common ones are digital signatures, symmetric encryption and asymmetric encryption. To prove the security of the cryptosystem, we must show that for any probabilistic polynomial-time adversary, the possibility of the attacker to break an cryptosystem is negligible as a function over the security parameter.

As is known, the most famous game-based security model is used for a digital signature scheme. Initially, the challenger generates an asymmetric key pair, the public key and private secret key respectively, and then the attacker may ask the challenger to produce signatures over his chosen messages m, which is known as chosen-ciphertext attack (CCA). Concurrently, the challenger believes that the attacker is a legitimated users and uses the signing algorithm and the private key faithfully. As a result, the adversary algorithm terminates and outputs (σ, m) to the attacker, in which σ is a signature over the chosen message m. It is no doubt that the attacker has broken the system if the verification algorithm declares that σ is a valid signature for the message m, although the attacker did not ask the challenger to sign this message. However, this is a strong notion of security, which cannot capture many capabilities of the attacker in reality.[29] Basically, the advantage of game-based security model is that it is simple and easy to understand and work with. Hence, our generic construction will be formulated on the basis of game-based security model.

3.4.2 Simulated-based Security Model

Besides, a simulation approach can also used to define a security model, in which an arbitrary probabilistic polynomial-time attacker and environments are involved in the cryptosystem. Moreover, it is also assumed that it exists an idealized cryptosystem that can never be broken and is also not practical. According to Dent's research paper[29], the cryptosystem will use an abstract third party truthfully to transport and/or vouch for data. Moreover, the operation of the trusted third party goes through without notice of both the environment and the attacker. By examining the difference of the outputs of environment and the attacker when they interact with the real cryptosystem and the idealized cryptosystem to determine whether a scheme is secure. The idea is that if the output of the environment and attacker when they interact with the idealised cryptosystem is as same as the output of interacting with the real cryptosystem, indicating that the real cryptosystem must be as secure as the idealised one. Basically, the cryptosystem must be secure if the probability of distinguishing the difference between the two outputs is neglible.[29]

Basically, simulation-based security models are stronger than game-based security models. Compared with game-based security models, simulation-based security models provide security proofs for the larger cryptosystem which will be used to provide more security guarantees. Similarly, simulation-based security models have also been used in realities. [29, 30]

To conclude, neither game-based security model nor simulation-based security model can provide full range of security guarantees and applicabilities for any cryptographic schemes. This project will make good use of Game-based model to develop a cryptographic security model for EMV, and may use the simulated proof system to prove the proposed construction.

4 A Theoretical Model

4.1 Basic Terminology

Symmetric Encryption

Basically, a symmetric-key encryption works using the following two transformations:

$$c = Enc_e(m);$$
$$m = Dec_d(c);$$

in which an encryption scheme is said to be symmetric if, given an encryption key and decryption key pair (e, d), it is computationally easy to determine e knowing only d (and vice-versa). Indeed, in many symmetric encryption schemes $e = d$. [9] Standards for symmetric ciphers are discussed in [31].

As is discussed, for each EMV card, it has a unique symmetric MK_{IC} derived from the Issuer's master key MK_I, which enables to provide support for securing transmission of personalization data used for EMV application and can also create application-level data for personalization. Moreover,MACs can be used to provide data integrity, origin authentication and confidentiality within the help of encryption, where the MAC computation is a DES-based CBC-MAC according to ISO/IEC 9797-1 and the encryption uses triple-DES Cipher Block Chaining (CBC). Hence, the EMV specifications require the encryption mechanism to use the block-ciphers in ECB or CBC mode. [11] Generally, the application symmetric secret keys (DES keys) enable the EMV transaction certificate generation, which must be created during the EMV card data preparation. The specified block Cipher is either DES or triple-DES, which are all discussed later.

Asymmetric Encryption

In general, asymmetric encryption is also known as public key cryptography, which replaces the use of identical keys for symmetric encryption (like DES/triple-DES) with two distinct keys, one public and one private (pk, sk), by/to an involved entity. The public key can be published or distributed in a directory, while the private key should keep secret. In an asymmetric encryption scheme, it works using the following transformations:

$$c = Enc_{pk}(m);$$
$$m = Dec_{sk}(c);$$

in which anyone who attempts to send a message m to the holder should firstly obtain the holder's public key to encrypt his message under this pk and then sends it to the holder of the corresponding private key. Consequently, only the holder of the private key can decrypt the encrypted message with sk. [32] Standards for asymmetric ciphers are discussed in [31].

For example, the use of RSA algorithm playing a significant role for EMV specifications. For the Issuer, it must have a public-private RSA key pair which can be used to enable EMV application after successful personalization. The Issuer's RSA key pair must be generated and certified by the Payment System Certification Authority, which shall be used in SDA/DDA/CDA. However, for each EMV card it must have a public-private RSA key pair if it supports the Offline Encryptrd PIN Verification.

Hash Functions

Generally, a standard hash functions is a function that takes an arbitrary finite bit-length of binary string as input (l), and then takes an infinite bit-length of binary string ($h(l)$) as output. Technically, the cryptographic hash function differs from the standard hash function, for the reason is that compared with the standard hash function, the cryptographic hash function should at least have the property of being one-way. More formally, the cryptographic hash function may have the following properties[32, 33]:

- **Preimage Resistant** It is computationally infeasible to find the corresponding unknown input, l', such that $h(l') = h(l)$, given all the pre-specified output. Hence, this property implies that it should be hard to find a message upon a given hash value.
- **Collision Resistant** It is computationally infeasible to find any two distinct inputs l and l' such that $h(l) = h(l')$. That is to say, it should be hard to find two messages upon the same hash value.
- **Second Preimage Resistant** It is computationally infeasible to find a second input l', such that l' differs from the given input l and $h(l) = h(l')$. In other words, given one message, it should be hard to find another message with the same hash value.

Nowadays, several cryptographic hash functions are widely used, and they are all iterative in nature. MD-5, RIPEMD-160 and SHA-1 are the most favorite hash functions among them. This project will only focus on SHA-1, which will be used for the signature scheme to create certificates, to sign the static data for SDA or sign data as part of DDA, according to the international standard ISO/IEC 9796-2. The specified Digital Signature Scheme gives message recovery mechanisms using a hash function. This signature scheme is based on the RSA algorithm and the hash-function SHA-1, as standardised in FIPS Pub 180-1 and ISO/IEC 10118-3.

Massage Authentication Codes(MACs)

Suppose there are two parties, who wished to send a message with each other by using a shared secret key. In order to protect against the data transmitted between the parties without being tampered with, they use the shared secret key (k) and a keyed algorithm $(f_k(x))$ to produce a check value sending along with the transmitted data. Such algorithms are called Message Authentication Codes, which are known as a family of functions providing (symmetric) data-origin authentication and message integrity. [9, 32]It can be defined as following: $MAC = f_k(m)$; in which MAC should have the following properties:[9, 33]

- **Ease of Computation:** Given a known function, $f_k()$, a key, k, and input, x, it should be computationally feasible to give $f_k(x)$
- **Compression:** The function $f_k()$ maps an infinite arbitrary bit-length as input, l, to a fixed bit-length of string as output, $f_k(x)$.
- **Computation Resistant:** Given zero or more text-MAC pairs, $(x_i, h_k(x_i))$, it is computationally infeasible to generate any text-MAC pair, $(x, h_k(x))$, for any new input x differs from xi.

Digital Signatures

Formally, a digital signature algorithm (DSA) is a kind of an asymmetric cryptographic scheme, also referring to as the DSS, which consists of a private signing procedure (using a private key) and a public verifying procedure (using the corresponding public key). [9] Standards for digital signatures are discussed in [31].

As mentioned above, the digital signature scheme for EMV specifications is based on the RSA algorithm and the hash-function SHA-1, as standardised in FIPS Pub 180-1 and ISO/IEC 10118-3. Consequently, RSA signatures will be the research aim of this project.

Public Key Certificates(PKI)

Public Key Infrastructure (PKI) is simply a means of reliably distributing public keys for use with asymmetric cryptography. Typically, a PKI involves one or more Certification Authorities (CAs) generating and distributing public key certificates. The use of a PKI requires the verifier of a certificate to have a trusted copy of the appropriate CA public key, and also a means of recognizing the security policy in force. [11]

In addition, PKI can be divided as open PKI and closed PKI, respectively. For an open PKI, it can only use for a single purpose, and operate under a single policy agreed by all parties. While for a closed PKI, it can be used generally, where the certificates may be issued and used under a number of different policies. It is clear that EMV uses a closed PKI, such as X.509 used for EMV certificates, which is a special certificate format to minimize the size of certificates.[11, 31]

4.2 Generic Construction

4.2.1 Entities Involved

Basically, the EMV specifications standardize the transactions between EMV-based IC card and the Merchant POS terminal, which the transactions involve a trusted group party, namely, *theCardholder,the Merchant, the Issuer and the Acquirer*. As specified before, the Cardholder is in possession of an EMV-compliant card which is authorized by the Issuer and then he/she can use this EMV card to perform a transaction at the merchant terminal. Because of the merchant terminal, it can communicate with the Cardholder's EMV card and can also build a communication over a secure channel with the Acquirer, who has to authorize and perform the actual transaction.

4.2.2 Formal Syntax and Algorithms

KEY INFRASTRUCTURE Reviewing the basis key infrastructure for each involved parties, here the key infrastructure for the generic construction still sticks with the basic key settings as specified in the cryptographic functionality for EMV specifications.(specified in section 2.3.1)

For the Issuer,it has a public-private key pair (P_I, S_I) and a master key MK_I, where the public component P_I is known by the Merchant terminal and the MK_I can be used for the IC card later. The following pseudo-code defines the key infrastructure for the Issuer:

$$\begin{aligned} S_I &= RSA\_GK_g(); \\ P_I &= RSA\_Pub\, S_I; \\ MK_I &= HMAC\_GK_g(); \end{aligned}$$

Basically, the Issuer's master key (MK_I)can be used in two ways: on the one hand, it can provide support for securing transmission of personalization data used for EMV application; on the other hand, it can create application-level data for personalization. Hence, the EMV personalization can only take place upon the initialization of MK_I and related specific data. After successful personalization, in order to enable EMV application, the Issuer's RSA key pair must be generated and certified by the Payment System Certification Authority, which shall be used in SDA/DDA/CDA.[34]
For each EMV card, it has a unique symmetric MK_{IC} derived from the Issuer's master key MK_I by encrypting Primary Account Number (PAN) and PAN sequence number (PANs). Hence, a session key SK can be derived from the EMV card's master key MK_{IC} by encrypting the card's Application Transaction Counter (ATC) and terminal-supplied nonce (UN = Unpredictable Number), which all can be defined as following:

$$\begin{aligned} create\_MK_{IC}\; pan &= Enc_{MK_I}(PAN, PANs); \\ create\_SK\; MK_{IC}\; atc &= Enc_{MK_{IC}}(ATC, UN); \end{aligned}$$

Generally, the application symmetric secret keys (DES keys) enable the EMV transaction certificate generation, which must be created during the EMV card data preparation. Since different MK_I keys are used for confidentiality, the derivation for the EMV card's MK_{IC} keys become diversity. Furthermore, if DDA/CDA is supported in the EMV card, then an EMV application-level RSA key pair must be generated and certified by the Issuer.[34] There may be one more pair RSA key pair required for the PIN decipherment if offline PIN encipherment is applied as mentioned before. The figure 4.2.1 and 4.2.2 illustrated the process of the derivation of MK_{IC} and SK. Consequently, the above process for each EMV card will be completed in the card initialization, which will be described as the ***Algorithm 4.1***:

```
// Card Initialisation
  if DDA/CDA is applied
     S_IC = RSA_GKg();
     P_IC = RSA_Pub S_IC;
  pan = mkNonce;
  MK_IC = create_MK_IC pan;
  SK = create_SK MK_IC atc;
```

```
    // preparation for actual transaction
    let (sda_enabled, dda_enabled, cda_enabled) =
            card_process (S_IC, P_IC, MK_IC, pan);
```

However, since the standard EMV specifications do not provide mechanisms to authenticate the merchant terminal, this project attempts to add a public key certificate for the merchant terminal, that is $Cert_M$, sending along with the signature of the merchant terminal to the Issuer.

ALGORITHMS AND QUERIES The developed cryptographic security model for EMV is specified as a tuple $SM_{EMV} = (CardTransactionInit, CardTransaction, Termianl-SDA, TerminalDDA, TerminalCDA, CardVerifyPIN)$ of polynomial-time algorithms whose intended usage and functionality are specified as follows.

$CardTransactionInit-$ In a setup phase of the EMV session, the Merchant terminal communicates with the EMV card to obtain corresponding information for the transaction. The EMV card also require the Merchant terminal to provide some information before indicating the features and the configurations about the card. As specified in [8], the EMV transaction begins with the Merchant terminal's SELECT_APPLICATION command. In response to the merchant terminal, the EMV card sends its response along with an empty Processing Options Objects List (PDOL), to specify which information is required from the merchant terminal by performing GET_PROCESSING_OPTIONS command, such as the Terminal Country Code and the purchase amount which will be authenticated in the next session. At the same time, the card constructs its Application Interchange Profile (AIP) and the Application File Locator (AFL). On the one hand, the AIP helps indicate its supported features (SDA/DDA/CDA/Cardholder Verification/Issuer Authentication) and whether the terminal risk management should be performed; On the other hand, the AFL provides identification of the files used in the transaction to indicate whether the EMV card support offline data authentication. Then the Merchant terminal compiles the READ_RECORD command to return the necessary Certification Authority identifier (CA) and public-key certificates, which uses to authenticate the card's public key in the $CERT_IC$ (certified by the Issuer's $CERT_I$).

Algorithm 4.2 ($CardTransactionInit$)

```
  //Perform initialization for the card transaction
    let card_process (S_IC, P_IC, MK_IC, pan) = force_online;
    SELECT_APPLICATION command;
    //send response with empty PDOL to the terminal
    select_application_response = pdol;
    //GET_PROCESSING_OPTIONS command from the terminal
    let pdol = get_processing_options ();
    //construct AIP and AFL and sends to the terminal
    let aip = (sda_enabled, dda_enabled, cda_enabled);
    let afl = "";
    READ_RECORD command;
    //send response to the terminal
    let response = read_record_response ((create_sdad (aip, pan)),
                   (create_cert S_I (P_IC, (sha1 (aip, pan))),pan));
```

$Terminal SDA-$ After the card initialization, the terminal obtain the AIP which indicates the mechanism that is supported by the EMV card for the data authentication. Once the EMV cards indicate the support for SDA, the Signed Static Application Data (SSAD) is created by the signed hash (SHA-1) of the concatenation of AFL files and the value of the AIP along with PAN. It is important to note that the API is an optional part depends on Static Data Authentication Object List (SDOL), which shall be specified in a standard record. Generally, as mentioned before, for the terminal, it stores the key pair of Certification Authority (CA): P_{CA}, S_{CA}, while for each EMV card, it stores the Issuer's public key certificate $CERT_I = Sign_{P_I}(S_{CA})$ and the Signed Static Application Data ($SSAD = Sign_{S_I}[SDA]$). To perform SDA, the terminal retrieves and authenticates the Issuer's public key (P_I) from $CERT_I$ by using P_{CA} and Static Application Data from SSAD by using P_I, based on the case study of Mitchell[11]. In addition, for the card supports SDA, the SSDA is required to retrieve in the card initialization, then no more communication is needed for SDA.

Algorithm 4.3 (**$Terminal SDA$**)

```
 //perform SDA if the EMV card supports and the
   transaction requests
   //create Signed Static Application Data for the card
   let create_sdad d = RSA_Sign S_I (sha1 (sad));
   if cda_enabled = false then
     if dda_enabled =false then
      if sda_enabled then begin
       let card_sda = sda_enabled (aip,afl,(P_I,S_I))
       //retrieve and authenticate P_I and
         Static Application Data
       let result_sda = RSA_Verify P_I (sha1 (aip, afl,
                        create_sdad d));
       end
      else
       return 0;
     else
      perform TerminalDDA;
    else
     perform TerminalCDA;
```

$Terminal DDA-$ However, if the EMV card supports DDA, there exists some additional communication. Initially, the certificate verification for the EMV card is performed similarly as the static data authentication in SDA. Secondly, the actual challenge-response authentication is performed by the Merchant terminal. For the challenge, an INTERNAL_AUTHENTICATE message is sent to the card, which contains Authentication Related Data (ARD) is specified by the Dynamic Data Objects List (DDOL). The DDOL always has a terminal-generated nonce (NonceT). In response to the challenge, the card generates a digital signature (Signed Dynamic Application Data) SDAD over the combination of the card's Dynamic Data and hash of this data by using the EMV card's S_{IC}. The card's Dynamic Data always contains a card-generated nonce (NonceC). Similarly, for the terminal, it stores the key pair of Certification Authority (CA): P_{CA}, S_{CA}, while for each EMV card, it stores the Issuer's public key certificate $CERT_I = Sign_{P_I}(S_{CA})$ and the card's public key certificate $CERT_{IC} = Sign_{S_I}(P_{IC})$. Then the terminal can retrieve and authenticate P_I and P_{IC} respectively.[8, 18, 11]

***Algorithm 4.4** (*$TerminalDDA$*)*

```
   //perform DDA if the EMV card supports and the
     transaction requests
    if cda_enableed = false then
     if dda_enabled then begin
      let card_dda = dda_enabled (atc, (P_I,S_I), NonceC);
      let NonceT = internal_authenticate data;
      let signature = RSA_Sign S_I (NonceC,NonceT);
      send internal_authenticate_response;
      let result_dda = RSA_Verify P_I (NonceC, NonceT,
                       signature);
      end
     else
      return 0;
    else
      perform TerminalCDA;
```

$TerminalCDA-$ Furthermore, a third variant method (Combined Data Authentication) CDA is also supported by the EMV specifications, which is essentially an enhanced vision of DDA. The difference is that CDA does not need additional messages as INTERNAL_AUTHENTICATE message required for DDA, while CDA for an card authentication is combined with the actual transaction. Similarly,the authentication for the Static Data will also use the certificate of the EMV card'd public key ($CERT_{IC}$) as with DDA. However, with CDA, the SDAD is a signature over a group of data,including the card-generated nonce (NonceC) and the terminal-generated nonce (NonceT) as with DDA, along with other specified data (Application Cryptograms (AC), Transaction Data Hash Code (TDHC)). [8]

***Algorithm 4.5** (*$TerminalCDA$*)*

```
  //perform CDA if the EMV card supports and the
    transaction requests
```

```
if cda_enableed = ture begin
  let card_dda = dda_enabled (atc, (P_I,S_I),
                NonceC, NonceT);
    let signature = RSA_Sign S_I (NonceC, AC, TDHC,
                       sha1 (NonceC, AC, TDHC, NonceT));
    let result_dda = RSA_Verify P_I (NonceC, NonceT,
                        signature);
    end
else
  return 0;
```

$CardVerifyPIN-$ Reviewing the Cardholder Verification discussed before, the EMV specifications specify the Cardholder Verification Methods (CVM) in several ways, including offline PIN verification (unencrypted and encrypted) and handwritten signature. (According to EMVCo[3], online PIN verification is performed at the discretion of the Issuer, which is not standardized in EMV specifications) For the reason of simplicity, this $CardVerifyPIN$ algorithm will only discuss the offline enciphered PIN verification in details. For the EMV card that supports offline verification by encrypted PIN, the connected Merchant terminal first requests a card-generated nonce (NonceC). By using the public key of the EMV card (P_{IC}), the Merchant terminal can encrypt the PIN along with the card-generated nonce, some random padding and an Unpredictable Number (UN). Then the terminal sends the result of the verification to the card, waiting for the card decrypts the encrypted string to check the PIN and the related random number it sent before.[11]

Algorithm 4.6 (**$CardVerifyPIN$**)

```
//perform offline encrypted PIN verification if requested
 let msg = card_pin_verify (atc, (S_IC, P_IC), NonceC);
 if card_pin_verify = get_command then
   Get the PIN pin;
   let msg = card_pin_verify Enc_P_IC (pin, NonceC,
             random padding, UN);
   Get verification response (Success/Failed);
 else
   return 0;
```

$CardTransaction-$ So far, after optional Cardholder Authentication and Cardholder Verification, this algorithm aims to perform the actual transaction. Typically, the EMV card supports both online and offline transaction based on the different application cryptograms, including an Application Authentication Cryptogram (AAC), an Authorization Request Cryptogram (ARQC),an Authorization Response Cryptogram (ARPC). If the EMV card and the Merchant terminal both agree on completing the actual transaction offline, the Merchant terminal sends a GENERATE_AC command (containing Transaction Data (TC))to the card to produce a cryptogram that can be verified by the Issuer later. Then the EMV card returns a Transaction Certificate (TC) to indicate the approval of this transaction. The response to the GENERATE_AC command can be specified in two categories: if no CDA is performed, in response to the terminal command, the card return TC which contains the CID, ATC and MAC. If either the Merchant terminal or the EMV card wants to perform the actual transaction online, the card first produces an Authorization Request Cryptogram (ARQC), which the Merchant terminal will forward it to the Issuer for approval, then after receiving the reply from the Issuer (ARPC)by using EXTERNAL_AUTHENTICATE command to indicate whether the transaction is approved or declined , then a TC or AAC will be generated by the card depending on the result of approval.[8, 35]

Algorithm 4.7 ($CardTransaction$ ***)***

```
//perform actual transaction
let card_transaction (atc, (S_IC, P_IC), SK, NonceC, aip,
                     cid, pdol, pan) = true;
let (sda_enabled, dda_enabled, cda_enabled) = aip;

if ac_type = ARQC begin
  let (ac_type, cda_requested) = generate_ac;
  //the Issuer returns a authorisation to indicate
    whether the approval success or fail
  EXTERNAL_AUTHENTICATE command;
  let ac_type = ARPC;
  if ac_type = TC begin
    GENERATE_AC command;
    let mac = MAC_{SK} (amount, terminal country code,
                        terminal verification results, date,
                        currency, transaction type, AIP, ATC);
    let TC = (cid, mac, atc);
   end
  else
     if ac_type = AAC begin
     GENERATE_AC command;
     let mac = MAC_{SK} (0);
     let AAC = (cid, mac, atc);
   end
end
else if ac_type = TC begin
  GENERATE_AC command;
  let mac = MAC_{SK} (amount, terminal country code,
                      terminal verification results, date,
                      currency, transaction type, AIP, ATC);
  let TC = (cid, mac, atc);
 end
else
```

```
    return 0;
```

4.3 Security Analysis

Basically, this section will examine to what extent the proposed cryptographic security model can meet the outlined EMV transaction security in section 3.2.

Authentication

We assumed that if the Merchant terminal performs the cardholder authentication successfully, it should be the highest card authentication method supported by both the card and the terminal, to rule out forced fall-back, e.g. from DDA to SDA. [8] By this assumption, the above model can meet the following all authenticity depends on the different involved entities.

1. *Entity authentication of the Cardholder to the Merchant terminal* This is accomplished by using PIN entry and DDA, where PIN verification can be performed either offline by the EMV card or online by the Issuer, and DDA authentication can be performed by the verification of $CERT_{IC}$.[21] As mentioned before, since online PIN verification is not standardized by EMV specifications, here the PIN verification in the following analysis refers to Offline PIN Verification. Moreover, the EMV specifications limit the time of unsuccessful PIN entries to reject an ambiguous transaction.

2. *Origin authentication to the Issuer for the payment instruction* The Issuer can use the $CERT_{IC}$ received after successful cardholder authentication, in order to verify the origin of the payment instruction (PI). In addition, during the transaction authorization, the ARQC and TC are generated using SK shared between the card and the issuer. [21]

3. *Entity authentication of the Cardholder to the Issuer* Similarly, this is also based on the Cardholder's PIN, where the Issuer assumed that PIN can only be known by the legitimate genuine Cardholder. This is reasonable based on the assumption that the Cardholder must trust the Merchant terminal when interacting with it. As a result, the Merchant terminal must display the correct transaction information to the Cardholder. Furthermore, it is also assumed that a fraudulent merchant is relatively easy to track and prosecute.[21]

Confidentiality

The confidentiality requirements for EMV are the usual ones, namely confidentiality of the private asymmetric keys and the shared symmetric keys, like RSA public-private key pair for Offline PIN verification. However, some transaction information is known for each entity. Hense, the confidentiality can not be fully met by the proposed security model.

1. *Confidentiality of PIN* As mentioned before, the PIN verification is done by the EMV card offline, which the Merchant terminal and the EMV card involved with a challenge-response before the Merchant terminal encrypting the PIN received via a secure PIN pad. Basically, the terminal uses either the card's RSA public key (P_{IC}) or the public key (P_{PE}) of the card's signature key pair to encrypt PIN entry and sends the encrypted PIN to the EMV card. Then the card decrypts the PIN by using the corresponding private key ($S_{IC} or S_{PE}$), in order to verify the PIN.[21]

2. *Confidentiality of the order information* Since the order information of each transaction is not encrypted on the card-terminal channel and the cardholder's information can be read by the terminal, the Issuer and the Acquirer. Hence, it remains more discussion for this confidentiality in the foreseeable future. [18, 21]

Integrity

1. *Integrity protection of the payment authorization* As mentioned, the use of MAC can provide integrity to AC, which is generated by the EMV card during the transaction authorization. Then the verification of MACs can be performed by using a shared secret key SK between the card and the issuer.[21]

Non-Repudiation

1. *Non-repudiation of origin for cardholder payment authorization messages sent to the issuer* The cardholder's correct PIN entry over the display of purchase price will trigger the generation of TC by the card. The evidence of sending TC to the issuer confirms the cardholder's authorization; however, since a TC is generated using a secret key shared by the issuer and the card, which means it can only provided limited non-repudiation property unless the origin of the data is combined with evidences to protect it against any attempt by the cardholder to denying sending the transaction related information, e.g. combined with audit trails.

5 An Research Result for the construction

5.1 The adversaries' capabilities – Oracles

Based on the analysis of the key infrastructure and the basic algorithms for EMV specifications, the correctness and security definitions for the EMV's cryptographic model can be formulated via a number of experiments. these experiments model an adversary's attack capabilities which provides the adversary with access to certain oracles.[36] As a matter of fact, different experiments will provide an adversary with different subsets of oracles. Hence, this project just introduce the oracles depending on some attacks referred in the previous section.

Basically, the oracles are specified and explained below based on the study of [36]. It is firstly assumed that the overlying experiment has run GKg for the Issuer and the EMV card to obtain the keys S_I, P_I, MK_I which are used by certain oracles. In addition, by the assumption that the following experiment will maintain the global variables, which are manipulated by the adversary's capability to access certain oracles: a set HC for the honest cardholders and a set CC for the corrupted cardholders, which are assumed initially empty; a table cpk such that $cpk[i]$ contains the public key of cardholder with identity ***i***.[36] (Meaning, it has the possibility that anyone can obtain an authentic copy of the personal public key for each EMV cardholder, which can be managed by PKI.)

$CrptCardholder(.)-$ By calling this *corrupt cardholder oracle* with argument an identity ***i***, the adversary can corrupt cardholder ***i*** and run its own GKg to set its personal public key (P_{IC}) to the value cpk. As a result, the oracle initializes the Issuer's state in preparation for the $CardTransactionInit$ algorithm. Then the adversary can derive its own MK_{IC} from the Issuer's MK_I.

```
$CrptCardholder(i)$

If $i \in HC \cup CC$ then return $\epsilon$
else $CC \leftarrow CC \cup i$
$cpk[i] \leftarrow (P_{IC})$
$dec^i \leftarrow cont$
$St^i{}_{iss} \leftarrow (MK_I, cpk[i])$
return 1
```

$CloningSDAC(.,.)-$ By calling this *clonding SDA card oracle* with having corrupted cardholder ***i***, the adversary can use this oracle in anticipation of engaging in the EMV transaction with the trusted Issuer and the honest Merchant terminal, in which the adversary plays the role of a cardholder to communicate with the terminal without necessarily executing the interacting algorithm. For the SDA cards can authenticate by presenting digitally signed data ($SSAD(cardnumber, othercarddata)$), it means that SDA cards cannot do asymmetric crypto. Besides, the lack of refreshing mechanism for SDA results in the certificate for PIN verification can be cloned for any offline transactions.Consequently, the adversary can perform the

$CardTransactionInit$ and provide the oracle with *i* and the transaction related data *rd* (sdad, aip, pan) to be sent to the merchant terminal. The oracle maintains the Issuer's state and receives a static certificate forwarded by the terminal after generally SDA card authentication request, then the terminal will always response "yes" to any PIN verification from the adversary.

```
$CloningSDAC(i)$
---
if $i \in CC$ then
$(St^i_{Iss}, Cert^i_{IC}, dec^i) \leftarrow (CardTransactionInit(i, sdad, aip, pan))$
$(i, dec^i) \leftarrow TerminalSDA(i, sdad)$ $response[i] \leftarrow CardVerifyPIN(i, Cert^i_{IC}, dec^i)$
if response = yes then return 1
else
return $\epsilon$
```

$CrptIssuer(.)$ After considering the adversary corrupted either by the cardholder or the terminal, it is necessary to think about the corruption of the Issuer. This oracle can be used by such an adversary in anticipation of engaging in the transaction protocol within an the honest cardholder and the Merchant terminal. Generally, the adversary plays the role of the Issuer and do not actually interact with the cardholder and the merchant before performing the actual card transaction. For DDA cards can authenticate by challenge-response, it means that DDA cards can perform both symmetric and asymmetric crypto without being cloned. However, the actual transaction is not tied to the card a authentication, hence, the terminal cannot spot the fake offline transactions performed by the fake Issuer. The adversary provides the oracle with *i* and the transaction related data *rd* (sdad, aip, pan) to be sent to the Issuer for running $CardTransaction$. The oracle then maintains the issuer's state and computes a response for the GENERATE_AC command, returning the outgoing messages to the adversary without performing the actual transaction with the trusted issuer.

```
$CrptIssuer(i)$
---
if $i \in CC$ return $\epsilon$
else
$(St^i_{Iss}, Cert^i_{IC}, dec^i) \leftarrow (CardTransactionInit(i, sdad, aip, pan))$
$(i, dec^i) \leftarrow TerminalSDA(i, sdad)$ $response[i] \leftarrow CardVerifyPIN(i, Cert^i_{IC}, dec^i)$
$generate_a c_r esponse[i] \leftarrow (St^i_{iss}, dec^i, GENERATE_A C command)$
return $M_{out}$
```

$CroptTerminal(.,.)$ In a certain circumstance, we will take the modified Merchant terminal into account, which means that the adversary has corrupted the terminal. By calling this *i* with having the corrupted Merchant terminal, the adversary can use this oracle in anticipation of engaging in the EMV transaction within the trusted Issuer and the honest Cardholder.

Meanwhile, the adversary plays the role of the terminal to communicate with the EMV card without necessarily executing their interacting algorithm. The honest cardholder *i* is processing his legitimate transaction in the modified Merchant terminal,which could not only perform the current transaction but forward all the transaction information to an accomplice to make the smartcard sensitive data go to a fake card. In this way, the adversary can purchase his transaction via the fake card and the modified terminal by forwarding his payment to the legitimate cardholder. The adversary provides the oracle with *i* and the transaction related data *rd* (sdad, aip, pan), initializing the personal public-private key pair for *i*($cpk[i], csk[i]$); then the oracle executes the transaction protocol session for the honest cardholder by running $TerminalSDA(TerminalDDA/TerminalCDA), CardVerifyPIN$ on behalf of the cardholder's EMV card authentication and PIN verification. Finally, the oracle returns the outgoing messages of these transaction algorithms to perform the adversary's "transaction" a before the actual transaction is completed.

```
$CrptTermianl(i, j)$

if $i \in HC$ then return $\epsilon$
$HC \leftarrow HC \cup i$
$(St^i_{IC}, (cpk[i], csk[i]), dec^i) \leftarrow (CardTransactionInit(i, sdad, aip, pan))$
$(i, dec^i) \leftarrow Terminal\ (i, sdad)$
$response[i] \leftarrow CardVerifyPIN(i, Cert^i_{IC}, dec^i)$
while $dec^i$ = cont and response[i] =yes do
if $j \in CC$ then
$(St^j_{IC}, (cpk[j], csk[j]), dec^j) \leftarrow (St^i_{IC}, (cpk[i], csk[i]), dec^i)$
$(j, dec^j) \leftarrow (i, dec^i)$
$response[j] \leftarrow CardVerifyPIN(j, Cert^j_{IC}, dec^j)$
return ($(j, St^j_{IC}, dec^j)$)
else return $\epsilon$
```

$CSK(.)-$ The adversary can call this *cardholder secret keys oracle* with argument the identity i of a cardholder to expose both the Issuer's private key $isk[i]$ and the cardholder's private key $csk[i]$. By obtaining ($isk[i], csk[i]$), the data sent between the EMV card and the Merchant terminal includes all information needed to make a fake magnetic stripe card.

```
$CSK(i)$

return ($isk[i], csk[i]$)
```

As specified, the above oracles conclude the possible queries that the adversary can make with each involved party based on the formal algorithms.

5.2 Primitives for generic construction

More formally, the research result will begin by describing the primitives used in our construction.

5.2.1 Digital Signature Schemes

The digital signature scheme used for EMV specifications can be specified as $\mathit{DS} = (KG_s, Sig, Vf)$, which represents an algorithm for key generation, signing and verifying, respectively. It should satisfy the standard notion of unforgeability under chosen message attack.[37]

Basically, consider the experiment $\mathbf{Exp}^{unforg-cma}_{\mathit{DS},A}(k)$ shown as following, which involves an adversary A, obtaining the public-private key pair (pk, sk) (eg.(P_{IC}, S_{IC}). Hence, the RSA signature scheme can be used for card authentication, transmitting certificates and establishing session keys during the transaction). This key pair (pk, sk) is generated by running KG_s over the security parameter $(pk.sk) \leftarrow KG_s(1^k)$. Then, the adversary A is given as input pk, and is also provided access to a signing oracle $Sig(sk, .)$. As a result, the adversary can sign any number of transaction messages of his own choice to $Sig(sk, .)$, in order to obtain in return signatures under secret key sk on his choosing transaction messages. Finally, the adversary is given as output (tm, σ), which is an attempted forgery on the choosing transaction messages.Generally, this experiment will return 1 if σ is a valid signature on tm, and tm was never queried to $Sig(sk, .)$;otherwise response 0 in return.[36] As a consequence, we can define the advantage of the adversary A as:

$$\mathbf{Adv}^{unforg-cma}_{\mathit{DS},A}(k) = Pr[\mathbf{Exp}^{unforg-cma}_{\mathit{DS},A}(k) = 1].$$

where the probability is taken on the coins of the algorithm KG_s for the key generation, Sig for the signature and Vf for verification, along with the coins of the adversary.

$\mathbf{Exp}^{unforg-cma}_{\mathit{DS},A}(k)$

```
$(pk.sk) \leftarrow KG(1^k)$
$(tm, \sigma) \leftarrow A(pk : Sig(sk, .))$
if the following are true then return 1
else return 0:
    -$Vf(pk, tm, \sigma) = 1$
    -A did not make oracle query tm
```

Consequently, we can say that a digital signature scheme is secure against adversaries under chosen message attack if the function $\mathbf{Adv}^{unforg-cma}_{\mathit{DS},A}(.)$ is negligible for any probabilistic polynomial time adversary A.

5.2.2 Symmetric Key Encryption Schemes

A symmetric encryption scheme used for EMV specifications can be specified as $\mathit{SE} = (KG_e, Enc, Dec)$, which represents algorithm for key generation, encryption and decryption. It should satisfy the standard notion of indistinguishability under adaptive chosen-ciphertext attack (IND-CCA) in the sense of in the sense of Rackoff and Simon.[38]

Consider the experiment $\mathbf{Exp}_{\mathit{SE},A}^{ind-cca}(k)$ shown as following, which involves an adversary A and formalized based on the "**find-then-guess**" definition of Bellare et.al..[39] For the **find** phase, the adversary A is given adaptive access to an encryption and decryption oracle, and then gives a pair of messages x_0, x_1 as output along with some state information s used for the **guess** phase. Then in the **guess** phase, the adversary A is given the encryption y of x_b and s, in which A must identity which of the two messages has been encrypted without querying the decryption oracle.We can define the advantage function of A as[40]:

$$\mathbf{Adv}_{\mathit{SE},A}^{ind-cca}(k) = Pr[\mathbf{Exp}_{\mathit{SE},A}^{ind-cca\_1}(k) = 1] - Pr[\mathbf{Exp}_{\mathit{SE},A}^{ind-cca\_0}(k) = 1].$$

where the probability is taken on the coins of the algorithm KG_e for the key generation, Enc for the encryption and Dec for the decryption, along with the coins of the adversary.

$\mathbf{Exp}_{\mathit{SE},A}^{ind-cca\_b}(k)$

```
    $r_e \leftarrow 0, 1_{r(k)}$
    $(x_0, x_1, s) \leftarrow A^{Enc_{se}^{re}, Dec_{se}^{re}}(\mathbf{find})$
    $y \leftarrow Enc_{r_e}(x_b)$
    $d \leftarrow A^{Enc_{se}^{re}, Dec_{se}^{re}}(\mathbf{guess}, y, s)$
    return d
```

Consequently, we can say that a symmetric encryption scheme SE is IND-CCA secure if the function $\mathbf{Adv}_{\mathit{SE},A}^{ind-cca\_b}(.)$ is negligible for any probabilistic polynomial time adversary A.

5.2.3 Public-Key Encryption Schemes

Similarly, a public-key encryption scheme used for EMV specifications can be specified as $\mathit{AE} = (KG_{ae}, Enc_{ae}, Dec_{ae})$, which represents algorithm for key generation, encryption and decryption. It should satisfy the standard notion of indistinguishability under adaptive chosen-ciphertext attack (IND-CCA).[41]

Consider the experiment $\mathbf{Exp}_{\mathit{AE},A}^{ind-cca}(k)$ shown as following, which involves an adversary A, obtaining the public-private key pair (pk, sk) (eg.the issuer uses a public-private RSA key pair to perform SDA/DDA/CDA, or the EMV card uses its RSA key pair to support offline encrypted PIN verification).In general, this key pair is generated by running the key generation algorithm randomly on the security parameter $(pk.sk) \leftarrow KG_e(1^k)$, in which the length of the randomness string $|r_e|$ is bounded by some fixed polynomial $r(k)$.[36] Then it is assumed that the adversary A can never query $Dec(sk, .)$ on a ciphertext returned by $Enc(pk, m_b)$. For a bit $b \leftarrow 0, 1$ and message m_b, the advantage function of A is defined as:

$$\mathbf{Adv}_{\mathit{AE},A}^{ind-cca}(k) = Pr[\mathbf{Exp}_{\mathit{AE},A}^{ind-cca\_1}(k) = 1] - Pr[\mathbf{Exp}_{\mathit{AE},A}^{ind-cca\_0}(k) = 1].$$

where the probability is taken on the coins of the algorithm KG_e for the key generation, Enc for the encryption and Dec for the decryption, along with the coins of the adversary.

$\mathbf{Exp}_{AE,A}^{ind-cca\_b}(k)$

```
$r_e \leftarrow 0, 1_{r(k)}$
$(pk.sk) \leftarrow KG_{ae}(1^k; r_e)$
$d \leftarrow A(pk, mb, Dec_{ae}(sk, .))$
return d
```

Consequently, we can say that a public-key encryption scheme AE is IND-CCA secure if the function $\mathbf{Adv}_{AE,A}^{ind-cca\_b}(.)$ is negligible for any probabilistic polynomial time adversary A.

5.2.4 Security model of MACs

Moreover, recall the cryptographic primitives, since MAC secure messaging can be used to ensure the integrity and confidentiality of commands sent by an issuer to the EMV card, a security model for MACs for EMV specifications can also be specified as $MA = (KG, T_k, V_k)$, which represents algorithm for key generation, MAC algorithm ($c = T_k(m)$) and verification algorithm which returns true or false ($T/F = V_k(m, c)$).It should satisfy the standard notion of selective forgery under chosen-message attack (SUF-CMA) or existential forgery under chosen-message attack (EUF-CMA).[42, 43]

Consider the experiment $\mathbf{Exp}_{MA,A}^{suf-cma}(k)$ shown as following, which involves an adversary A, obtaining the ability to unrestrictedly run T_k and V_k given access to the MAC-generation oracle O_T and verification oracle O_V (eg. MAC secure messaging used for EMV data integrity and confidentiality). Eventually, the adversary A outputs a message/tag pair (m,c), where such that $V_k(m, c) = 1$ and $T_k(m)$ would never return c.[42]Hence, the advantage of the adversary is defined as:

$$\mathbf{Adv}_{MA,A}^{suf-cma}(k) = Pr[\mathbf{Exp}_{MA,A}^{suf-cma}(k) = 1]$$

where the probability is taken on the coins of the algorithm KG for the key generation, T_k for the MAC-generation and V_k for the verification, along with the coins of the adversary.

$\mathbf{Exp}_{MA,A}^{suf-cma}(k)$

```
$k \leftarrow KG(1^k)$
$(m, c) \leftarrow A(k, T_k, V_k)$
if the following are true then return 1
else return 0:
    -$V_k(m, c) = 1$
    -$T_k$ would never return c
```

As a result, we can say that a security model of MACs MA is SF-CMA secure if the function $\mathbf{Adv}_{MA,A}^{sf-cma}(.)$ is negligible for any probabilistic polynomial time adversary A. Furthermore, if the adversary's message m had not been queried previously, he can still win if c was never returned by $T_k(m)$, which can satisfy the standard notion of existential forgery under chosen-message attack (EUF-CMA). Then the advantage of the adversary is defined as:

$$\textbf{Adv}^{euf-cma}_{\textbf{\textit{MA}},A}(k) = Pr[\textbf{Exp}^{euf-cma}_{\textbf{\textit{MA}},A}(k) = 1 \cup NonQueriedEarlier]$$

where $NonQueriedEarlier$ means that the adversary A outputs a message m that has not been queried previously to the MAC-generation oracle O_T. Any stateless, deterministic MAC satisfies SUF-CMA whenever it satisfies EUF-CMA.[43, 44]

5.2.5 Simulation-sound Non-Interactive Zero Knowledge Proof System

More formally, a simulation-sound Non-Interactive Zero Knowledge proofs of cardholder ownership in NP languages are still necessary. The basic terminology for a simulation-sound NIZK proof systems can be specified as following. According to Bellare, et.al[36]:

> *"An NP-relation over domain Dom $\subseteq \{0,1\}^*$ is a subset ρ of $\{0,1\}^* \times \{0,1\}^*$ such that ownership of $(\chi, \omega) \in \rho$ is decidable in time polynomial in the length of the first argument for all χ in domain Dom. The language associated to ρ is the set of all $\chi \in \{0,1\}^*$ such that there exists a ω for which $(\chi, \omega) \in \rho$. Often we just use the term NP-relation, the domain being implicit. If $(\chi, \omega) \in \rho$ we will say that χ is a theorm and ω is a proof of χ"*(quoted in [36])

Initially, we say that a *(P,V)* is a non-interactive proof system for ρ over Dom if the exsited polynomials *p* and *l* can satisfy the following two conditions[45]:

- **Perfect Completeness:** $\forall k \in N, \forall(\chi, \omega) \in \rho$ *with* $|\chi| \leq l(k)$ *and* $\chi \in$ *Dom,* then for all adversaries we have :

$$\Pr[\sigma \leftarrow \{0,1\}^{p(k)}; (\chi, \omega) \leftarrow A(\sigma); \pi \leftarrow P(\sigma, \chi, \omega) : V(\sigma, \chi, \pi) = 1 \text{ if } (\chi, \omega) \in \sigma] = 1$$

- **Perfect Soundness:** $\forall k \in N, \forall \hat{P}, \forall \chi \in$ *Dom such that* $\chi \notin L_\rho$, then for all adversaries we have:

$$\Pr[\sigma \leftarrow \{0,1\}^{p(k)}; (\chi, \pi) \leftarrow A(\sigma) : V(\sigma, \chi, \pi) = 0 \text{ if } \chi \notin L_\rho] = 1$$

Hence, a non-interactive proof system (P, V) for relation ρ (eg.a simulator) can be formulated as a polynomial time algorithm running in two stages. For the randomized **gen** stage it produces a simulated commone reference string σ, meanwhile for the **prove** stage it inputs a *theorm* χ and states information passed on by the first stage, and then outputs a simulated proof for the validity of χ with respect to σ.[36, 45]

Then, consider the zero-knowledge requirement based on the above non-interactive proof system (P, V) for relation ρ bounded with a simulator SIM. Basically, Zero-Knowledge is defined by means of a *distinguisher* D which tries to distinguish between proofs produced by a prover or a simulator, which represents a real common random string and simulated common random string, respectively. The following two experiments ($\textbf{Exp}^{zk\_0}_{P,SIM,D}(k)$ and $\textbf{Exp}^{zk\_1}_{P,SIM,D}(k)$) show an involved *distinguisher* which is required to supply or loses a correct witness for χ relative to ρ.[36]

$\mathbf{Exp}^{zk\_0}_{P,SIM,D}(k)$

```
    $(\sigma, St_S) \leftarrow SIM(textbfgen, \{0,1\}^{p(k)})$
    $d \leftarrow D(\sigma : \mathbf{Prove}_0(.,.))$
    return d
$Prove_0(\chi, \omega)$
    $\pi \leftarrow SIM(\mathbf{prove}, St_S, \chi)$
    $return \pi$
```

In the experiment $\mathbf{Exp}^{zk\_0}_{P,SIM,D}(k)$, it is assumed that a simulated common random string σ is produced via the simulator's randomized **gen** algorithm.Then, the *distinguisher* D choose a *theorm* χ based on σ. By quering its $\mathbf{Prove}_0$ oracle with $(\chi, \omega) \in \rho$ where $\chi \in$ Dom, D is obtained as challenge a proof π, which is produced by the simulator's **prove** algorithm. However, for the experiment $\mathbf{Exp}^{zk\_1}_{P,SIM,D}(k)$, it is assumed that a real common random string σ is produced randomly ($\sigma \leftarrow \{0,1\}^{p(k)}$). Moreover, By quering its $\mathbf{Prove}_1$ oracle with $(\chi, \omega) \in \rho$ where $\chi \in$ Dom, D is obtained as challenge a proof π, which is produced by the Prover P.[36]

$\mathbf{Exp}^{zk\_1}_{P,SIM,D}(k)$

```
    $\sigma \leftarrow \{0,1\}^{p(k)}$
    $d \leftarrow D(\sigma : \mathbf{Prove}_1(.,.))$
    return d
$Prove_0(\chi, \omega)$
    $\pi \leftarrow P(\{0,1\}^{p(k)}, \chi, \omega, \sigma)$
    $return \pi$
```

Concurrently, the advantage of *distinguisher* D can be defined when it is assumed that D can make exactly one query to $\mathbf{Prove}_b$:

$$\mathbf{Adv}^{zk}_{P,SIM,D}(k) = Pr[\mathbf{Exp}^{zk\_1}_{P,SIM,D}(k) = 1] - Pr[\mathbf{Exp}^{zk\_0}_{P,SIM,D}(k) = 1]$$

Formally, we say that a non-interactive proof system *(P,V)* is computational zero-knowledge if there exists a polynomial time simulator SIM such that for any probabilistic polynomial time *distinguisher* D, the function $\mathbf{Adv}^{zk}_{P,SIM,D}(.)$ is negligible.[36, 46] Consequently, we say that *(*P,V,SIM) is a non-interactive zero-knowledge proof system based on the dependency of SIM on *(P,V)*.

For the property of simulation soundness, it can be defined by using the experiment $\mathbf{Exp}^{ss}_{\Pi,A}(k)$ within a simulation-soundness adversary A, where let $\Pi = ($P,V$, SIM)$ be a zero-knowledge interactive proof system for NP-relation ρ over domain Dom.[45] Initially, the random string is similar to the simulated common random string in the experiment $\mathbf{Exp}^{zk\_0}_{P,SIM,D}(k)$, which is also generated by running the simulator's **gen** algorithm. Then, the string σ is passed to the involved similation-soundness adversary given the access to the oracle $SIM(\textbf{prove}, St_S, .)$. At the same time. it is also necessary to assume that the simulation-soundness adversary can make exactly one query to $SIM(\textbf{prove}, St_S, .)$. The experiment ends with outputting a pair (χ, π).[36]

$\textbf{Exp}^{ss}_{\Pi,A}(k)$

$(\sigma, St_S) \leftarrow SIM(\textbf{gen}, \{0,1\}^{p(k)}); (\chi, \pi) \leftarrow A(\sigma : SIM(\textbf{prove}, St_S, .))$
If all of the following are true then return 1 else return 0:

(1) $\chi \notin L_\rho$

(2(π was not returned by A's oracle in response to a query χ

(3) $V(\{0,1\}^{p(k)}, \chi, \pi, \sigma)$

Hence, the advantage of the simulation-soundness adversary is defined by

$$\textbf{Adv}^{ss}_{\Pi,A}(k) = Pr[\textbf{Exp}^{ss}_{\Pi,A}(k) = 1]$$

then we say that $\Pi = (P, V, SIM)$ is a simulation-soundness system for any probabilistic polynomial time adversary A, if there exists a negligible function $\nu_A(.)$ such that $\textbf{Adv}^{ss}_{\Pi,A}(k) \leq \nu_A(k)$ for all k.[36]

5.3 Insight into generic construction

Basically, it fixed a digital signature scheme *DS* $= (KG_s, Sig, Vf)$, a symmetric encryption scheme *SE* $= (KG_{se}, Enc_{se}, Dec_se)$, a public-key encryption scheme *AE* $= (KG_{ae}, Enc_{ae}, Dec_{ae})$ and MACs $MA = (KG, T_k, V_k)$. Then the building blocks above can be used to construct a more formalised security model for EMV specifications, based on the previous generic construction $SM_{EMV} =$ (*CardTransactionInit, CardTransaction, TerminalSDA, TerminalDDA,CardVerifyPIN*) of polynomial-time algorithms, that can satisfy data authentity, integrity, confidentiality and non-repudiation.

The issuer's public-private key pair (P_I, S_I) consists of the security parameter k, a public RSA encryption key P_I (in order to authenticate the EMV card's public key P_{IC} in $Cert_{IC}$) and a verification key S_I for verifying the issuer's digital signature on $Cert_{IC}$, which we call the *certificate verification* key. And the issuer's symmetric master key is generated by MACs. And two reference strings R_1 and R_2 for NIZK proofs.

In the EMV card transaction initialisation phase, the issuer's verification key P_I and the corresponding siging key S_I are generated and verified by the Payment System Certification Authority. The issuer uses its personal private S_I to produce a signature Sig_I on $Cert_{IC}$. The signature Sig_{S_I} prevents the cardholder from being fraud by a corrupt issuer. Then the issuer sends P_I, Sig_{S_I} to the merchat terminal, who attemps to verify the EMV card's publick key P_{IC} in $Cert_{IC}$ (performed by $TerminalSDA$) and first verifies the issuer's public key in $Cert_I$ by using its copy of CA public key. Then the terminal stores (P_I, Sig_{S_I}), which later can be used for P_{IC} verification.

Moreover, in the initialisation phase, the EMV card produce a signature for the basic information about itself (eg. the cardholder's account details, the information supported by the card and their configurations, specified as ($CA, Cert_I, Cert_{IC}, PAN, ATC, ...$))under MK_{IC}, which is a unique symmetric key derived from the issuer's MK_I by encrypting PAN and PANs. To make the this verifiable without losing integrity and confidentiality, it MACs the issuer's MK_I and then proves in zero-knowledge that verification succeeds with respect to P_I.A session key SK then can be derived from the EMV card's MK_{IC} by encryptinh the card's ATC and UN. Moreover, in order to prevent the adversary from simply cloning the legitimate cardholder's key pair for using abroad , it also encrypts $(Cert_{IC}, Cert_I)$ and proves in zero-knowledge that this certificate is a signature of the genuine cardholder under the certificate verification key $(P_I, S_I, P_{IC}, S_{IC})$.Then, the key verification for EMV cards comes down to verification of the NIZK proofs. By recalling the ***Algorithm 4.1*** and ***Algorithm 4.2***, the following provides a detailed specification of the ***Algorithm CardTransactionInit***:

```
Algorithm CardTransactionInit
    $R_1 \leftarrow \{0,1\}^{p_1(k)}; R_2 \leftarrow \{0,1\}^{p_2(k)}$

    $r_e \leftarrow \{0,1\}^{r(k)}$
    $(P_I, S_I) \leftarrow KG_{ae}(1^k; r_e)$
    $(P_{IC}, S_{IC}) \leftarrow KG_{ae}(1^k)$
    $MK_I \leftarrow MAC(KG_{se}(1^k))$
    $MK_{IC} \leftarrow Enc_{ae}^{MK_I}(PAN, PANs)$
    $SK \leftarrow Enc_{ae}^{MK_{IC}}$
    $SAD \leftarrow Hash_{SHA-1}(aip, afl, PAN)$
        $SSAD \leftarrow Sig_{S_I}(SAD)$ (optional, only for SDA)
    return $(P_I, Sig_{S_I}(Cert_{IC}), P_{IC}, Sig_{SK}(SSAD)(optional))$
```

As discussed before, the merchant terminal retrieve and authenticate the Issuer's public key (P_I) from $CERT_I$ by using P_{CA} and Static Application Data from SSAD by using a digital signature scheme under (P_I, S_I)[11]. In addition, for the EMV card supports SDA, since the SSAD is required to retrieved in the initialization, there is no more communication is needed for SDA. By recalling the ***Algorithm 4.3***, the following provides a detailed specification of the ***Algorithm TerminalSDA***:

```
Algorithm TerminalSDA

    if sda_enabled = true
        $card\_sda \leftarrow (aip, afl, SSAD, (P_I, S_I))$
        return $result\_sda \leftarrow Vf_{P_I}(Hash_{SHA-1}(aip, afl, SSAD))$
    else
        return 0
```

However, if the EMV card supports DDA, there exists some additional communication. As mentioned in section 2.2.1:

”the Merchant terminal first sends an INTERNAL _AUTHENTICATE command (IAC) which contains an unpredictable number to the EMV card. Then the EMV card digitally signs the (IAC) data and sends it along with the Issuer’s $CERT_I$ and the card’s $CERT_{IC}$ to the Merchant terminal. When the Merchant terminal recieves the data, it uses its stored copy of top-level P_{CA} to retrieve the Isser’s P_I and then the Issuer’s P_I can be used to verify the $CERT_{IC}$ to obtain the card’s P_{IC}. finally, the EMV card’s public key P_{IC} can then be used to verify the ”dynamic” signature created by the card before. Within the terminal-generated nonce (unpredictable number)”

By performing DDA within the dynamic signature, it can prevent the EMV card being cloned, consequently, the following provides a detailed specification of the ***Algorithm TerminalDDA*** based on the ***Algorithm 4.4***:

Algorithm TerminalDDA

```
if dda_enabled = true
    $card\_dda \leftarrow (atc, (P_I, S_I, NonceC))$
    $NonceT \leftarrow$ internal_authenticate data
    dynamic signature $\sigma_1 \leftarrow Sig_{S_I}(NonceC, NonceT)$
    return $result\_dda \leftarrow Vf_{P_I}(NonceC, NonceT, \sigma_1)$
else
    return 0
```

Besides, the EMV card can also support a third variant method CDA, which is regarded as an enhanced vision of DDA. The main difference is that CDA do not need an **INTERNAL_AUTHENTICATE** message to specify NonceT. Meanwhile, to perform CDA, a signature over a group of data is needed, including the card-generated nonce (NonceC) and the termianl-generated nonce (NonceT) as with DDA, along with other specified data (Application Cryptograms (AC), Transaction Data Hash Code (TDHC)).By recalling the ***Algorithm 4.5***, the following provides a detailed specification of the ***Algorithm TerminalCDA***:

Algorithm TerminalCDA

```
if cda_enabled = true
    $card\_cda \leftarrow (atc, (P_I, S_I), NonceC, NonceT)$
    signature $\sigma_2 \leftarrow Sig_{S_I}(NonceC, AC, TAHC, Hash_{SHA-1}(NonceC, AC, TDHA))$
    return $result\_cda \leftarrow Vf_{P_I}(NonceC, NonceT, \sigma_2)$
else
```

```
    return 0;
```

After successful card initialisation and cardholder authentication, the cardholder verification process is invoked. It is known that the EMV specifications specify the cardhodler verification methods in several ways. Here, the discussion will follow the generic construction for cardholder verification, that is offline encrypted PIN verification.As specified before,

> *”By using the public key of the EMV card (P_{IC}), the Merchant terminal can encrypt the PIN along with the card-generated nonce, some random padding and a Unpredictable Number (UN). Then the termianl sends the result of the verification to the card, waiting for the card decrypts the encrypted string to check the PIN and related random number it sent before.”*

Hence, by recalling the ***Algorithm 4.6***, the following provides a detailed specification of ***Algorithm CardVerifyPIN***:

Algorithm CardVerifyPIN

```
$K \leftarrow (1^k, R_1, R_2, P_IC, MK_{IC})$
$m \leftarrow Vf_K(atc, (P_{IC}, S_{IC}), NonceC)$
if card_pin_verify = get_command
    pin ← PIN
    $C \leftarrow Enc_{ae}^{P_{IC} \backslash P_{PE}}(pin, NonceC, R_1, R_2, UN)$
    $s \leftarrow Sig(S_{IC}, m)$
    $\pi \leftarrow P(1^k, (P_{IC}, m, C), (Cert_{IC}, s), R_1, R_2)$
    $\sigma \leftarrow (C, \pi)$
    return $\sigma$

                                        $\longrightarrow$
                                             if $V(1^k, (P_{IC}, Dec_{ae}^{S_{IC} \backslash S_{PE}}(m, \sigma)),$
                                                 $\pi, R_1, R_2)$ = 0
                                                 return 0
                                             else
                                                 return 1
                                        $\longleftarrow$

else
    return 0
```

Finally, after successful cardholder authentication and cardholder verification, in order to perform actual transaction for EMV cards. As is known, the EMV card supports both online and offline transactions based on the different application cryptograms, which are usually specified as (TC/AAC/ARQC). Formally, the cryptogram TC is produced in response to the Merchant Ternimal’s **GENERATE_AC** command when the actual transaction aims to be completed offline, where TC is a MAC computed over the transaction details by using the share symmetric

SK between the EMV card and the issuer. However, if the actual transaction aims to perform online, the cryptogram ARQC is produced before a TC or AAC will be generated depending on the result of ARPC. by recalling the ***Algorithm 4.7***, the following provides a detailed specification of ***Algorithm CardTransaction***:

Algorithm CardTransaction

```
$\iota \leftarrow (atc, (P_{IC}, S_{IC}), SK, NonceC, NonceT, aip, cid, pdol, pan)$
$\iota$ = true
if $ac\_type$ = ARQC
    (ac_type,cda_requested) $\leftarrow$ (GENERATE_AC(rcp,td),EXTERNAL_AUTHENTICATE command)
    $ARQC \leftarrow T_{SK}(td)$
    ac_type $\leftarrow$ ARPC
    if ac_type = TC
        $iad \leftarrow T_{SK}(ARQC, NonceC)$
        $td \leftarrow (amount, currency, date, pan, pdol, NonceT)$
        $mac \leftarrow T_{SK}(iad, td, aip, atc)$
        $TC \leftarrow (cid, mac, atc)$
    else
        if ac_type = AAC
            $AAC \leftarrow (aid, atc)$
else if ac_type = TC
    $iad \leftarrow T_{SK}(ARQC, NonceC)$
    $td \leftarrow (amount, currency, date, pan, pdol, NonceT)$
    $mac \leftarrow T_{SK}(iad, td, aip, atc)$
    $TC \leftarrow (cid, mac, atc)$
else
    return 0
```

6 Conclusion and Future work

6.1 Specification of generic construction

Basically, one would expect given the previous modeling results like the following be proved. The specification of the cryptographic security model for EMV begins with the witness ralation ρ underlying the zero-knowledge proofs. Relation ρ can be defined as follows:$((P_I, P_{IC}, SK, m, C), (MK_I, MK_{IC}, Cert_I, Cert_{IC}, s, r), (\sigma, mac)) \in \rho$ iff

$$Vf(P_I, Cert_I) = 1; Vf(P_{IC}, S_{IC}) = 1; Vf(MK_I, MK_I C, m, s) = 1 \text{ and}$$
$$V(m, s, \sigma) = 1; Enc(P_{IC}, r) = C \text{ and}$$
$$V_k(SK, mac) = True$$

Here, $Vf(P_I, Cert_I) = 1$ and $Vf(P_{IC}, S_{IC}) = 1$ means the success of cardholder authentication, $Vf(MK_I, MK_I C, m, s) = 1; V(m, s, \sigma) = 1$ and $Enc(P_{IC}, r) = C$ implies the completion of cardholder verification by using offline encrypted PIN verification methods. Finally, $V_k(SK, mac) = True$ illustrates the appoval of the actual transaction by online or offline, in which using coins r and assume that $|r| = k$. Then, the domain Dom corresponding to ρ is the set of $((P_I, P_{IC}, SK, m, C)$ where the public key for the issuer and the EMV card (P_I, P_{IC}) has obtained non-zero probability under the security parameter k. Consequently, ρ becomes an NP relation over Dom.

Informally,the fixed digital signature scheme *DS* $= (KG_s, Sig, Vf)$, a symmetric encryption scheme *SE* $= (KG_{se}, Enc_{se}, Dec_s e)$, a public-key encryption scheme *AE* $= (KG_{ae}, Enc_{ae}, Dec_{ae})$ and MACs $MA = (KG, T_k, V_k)$ along with NP-relations ρ over domain Dom and their non-interactive proof system (P, V) as above, the specification of $SM_{EMV} =$ (*CardTransactionInit, CardTransaction,TerminalSDA, TerminalDDA, CardVerifyPIN*) of polynomial-time algorithms denote the cryptographic security model for EMV.

The following lemmas can be derived as the main result for this generic construction:

Lemma 6.1. *If AE and SE are both IND-CCA secure encrypting schemes, DS is secure against forgery under chosen-message attack and (P, V) is a simulation sound, computational zero-knowledge proof system for ρ over domain Dom, then the cryptographic security model for EMV SM_{EMV} is authenticity.*

According to the study of Bellare,et.al.[36], by the assumption that P is computational zero-knowledge for ρ over domain Dom, then a simulator SIM such that $\Pi = (P, V, SIM)$ is a simulation sound zero-knowledge non-interactive proof system for L_ρ. Now, we can show that for any probabilistic polynomial time IND-CCA adversary B mounting an attack against authenticity of SM_{EMV},(for example the adversary use cloned SDA cards for offline transactions, which has specified in oracle $CrptCardholder(.)$), one can construct polynomial time IND-CCA adversaries A_0,A_1 attacking encryption scheme *AE* and *SE* and A_2 attacking digital signature scheme *DS*, and an adversary A_s against the simulation soundness of Π along with a distinguisher D distinguishes the simulated proofs from the real proofs. Then for all $k \in N$

$$\mathbf{Adv}^{auth}_{SM,B}(k) \leq$$

$$\mathbf{Adv}^{ind-cca}_{AE,A_0}(k) + \mathbf{Adv}^{ind-cca}_{SE,A_1}(k) + \mathbf{Adv}^{unforg-cma}_{DS,A_2} + \mathbf{Adv}^{ss}_{\Pi,A_s} + 2.\mathbf{Adv}^{zk}_{P,SIM,D}$$

Based on the assumption on the security of the building blocks of the construction of cryptographic security model for EMV, it is known that the function on the right side of the above inequation are all negligible so that the function in the left side is also negligible, which demonstrate that the construction SM_{EMV} is authenticity.

Lemma 6.2. *If DS is secure against forgery under chosen-message attack, MA is secure against existential forgery under chosen-message attack and (P, V) is a simulation sound, computational zero-knowledge proof system for ρ over domain Dom, then the cryptographic security model for EMV SM_{EMV} are integrity and confidentiality.*

Similarly, by the assumption that P is computational zero-knowledge for ρ over domain Dom, (P, V) is a sound proof system for ρ. Then, we can show that for any probabilistic polynomial time adversary B mounting an attack against integrity and confidentiality of SM_{EMV},(for example the adversary fool a terminal into thinking a PIN was correctly entered, which has specified in oracle $CrptTerminal(.,.)$), one can construct polynomial time adversaries A_1 attacking digital signature scheme *DS* and A_2 attacking security model for MACs. Then for all $k \in N$

$$\mathbf{Adv}^{int,con}_{SM,B}(k) \leq 2^{-k} + \mathbf{Adv}^{unforg-cma}_{DS,A_1}(k) + \mathbf{Adv}^{euf-cma}_{MA,A_2}(k)$$

Based on the assumption that the digital signature scheme *DS* and security model for MACs *MA* are secure, it follows that he function on the right side of the above inequation are all negligible so that the advantage function in the left side is also negligible, which demonstrates that the construction SM_{EMV} are integrity and confidentiality.

Lemma 6.3. *If DS is secure against forgery under chosen-message attack, MA is secure against existential forgery under chosen-message attack and (P, V) is a simulation sound, computational zero-knowledge proof system for ρ over domain Dom, then the cryptographic security model for EMV SM_{EMV} are non-repudiation.*

Meanwhile, let B a non-repudiation adversary B mounting an attack against SM_{EMV}, who aims to deny the purchased transaction of EMV cards. By the assumption that P is computational zero-knowledge for ρ over domain Dom, (P, V) is a sound proof system for ρ. Then, one can construct polynomial time adversaries A_1 attacking digital signature scheme *DS* and A_2 attacking security model for MACs. Then for all $k \in N$

$$\mathbf{Adv}^{nr}_{SM,B}(k) \leq 2^{-k+1} + \mathbf{Adv}^{unforg-cma}_{DS,A_1}(k) + \mathbf{Adv}^{euf-cma}_{MA,A_2}(k)$$

Based on the assumption that the digital signature scheme *DS* and security model for MACs *MA* are secure, it follows that he function on the right side of the above inequation are all negligible so that the advantage function in the left side is also negligible, which demonstrates that the construction SM_{EMV} are non-repudiation.

In the above two lemmas, since (P, V) is assumed to be a sound proof system for ρ, then for any probabilistic polynomial time adversary, the probability of $V(m, s, \sigma) = 1$ can be defined as below:

$Pr[((P_I, P_{IC}, SK, m, C), (MK_I, MK_{IC}, Cert_I, Cert_{IC}, s, r), (\sigma, mac)) \notin L_\rho \cup V(m, s, \sigma) = 1]$

$\leq 2^{-k}$

For the adversary needs to succeed a transaction purchase before denying his or her transaction details, then the probability becomes

$Pr[((P_I, P_{IC}, SK, m, C), (MK_I, MK_{IC}, Cert_I, Cert_{IC}, s, r), (\sigma, mac)) \notin L_\rho \cup V(m, s, \sigma) = 1]$

$\leq 2^{-k+1}$

for a non-repudiation adversary.

Since this projects aim was to develop the previous security model for EMV, we hope that future projects will be able to formally prove the above results.

6.2 Critical evaluation of the project

The aims and objectives set in the beginning of this dissertation were to review the literature of EMV specifications to help understand the current advances in this area, and on the basis of the cryptographic primitives to design a security model for EMV transactions. By observing the security property and threats for EMV specifications, a theoretical game-based security model has been constructed to capture the adversary's capability. Roughly, this project not only achieves what it was set to do, but it also goes beyond to research and proof the security requirements of our generic construction by the assumption of the involved cryptographic schemes secure against the corresponding attack.

Basically, the project does cover all the important options for the EMV cards and the Merchant Terminal, where it provides all three types of card authentication methods (SDA,DDA,CDA) and two transaction types(online,offline). From the point of general cryptographic view, a research result examined the generic construction for EMV specifications, where it concluded that the construction can meet the security requirements for EMV specifications if the involved cryptographic schemes were secure. In fact, the cryptographic security model for EMV can capture the possibility of some known weakness and limitations in the security of EMV, for example:

- using cloned SDA cards to perform offline transaction
- using incorrect PIN to fool the merchant terminal
- using DDA cards to repudiate or fake transaction

According to Ruiter, et.al[8], these problems are inevitable when someone is trying to establish the very generic security requirement for the EMV transaction. Since when a transaction is performed completely, the involved parties reach an agreement on the transaction parameters, including the card authentication mechanisms, the PIN verification process and the transaction types and so on.

However, given the complexity of EMV specifications, the constructed cryptographic security model for EMV is still surprisingly small. Certainly, the construction abstracts from the more lower-level details which are specified in more than 700 pages EMV specifications. It seems that such a abstraction for RMV specifications is crucial to understand it as a whole. Hence, except for the EMV cards and the Merchant Terminal, we can also extend this cryptographic security model for EMV to conclude the Issuer. The reason is that within the Issuer the transaction details cannot only be verified by both the EMV cards itself and the Merchant Terminal, but also includes the agreement of the Issuer. Typically, it is a more interesting example for the widely-used online transaction, where the agreement of the Merchant Terminal depends on the Issuer's agreement over this transaction. From this point of view, our generic construction is not perfect enough to put it in reality.

Besides, because of the limitations of time and knowledge, the syntax and algorithms used for constructing the cryptographic security model for EMV are illustrated based on the pseudo-code, which might be improved to be more effective and accurate in the near future, as the increasing of researching and understanding of the related cryptographic knowledge and the public scrutiny of the EMV specifications.

6.3 Conclusion and Future work

Basically, this dissertation presents a comprehensive research on the security of EMV specifications and provides a theoretical cryptographic model for securing EMV transactions. On the other hand, there are still some open questions about several aspects of the underlying security of the EMV specifications itself and the feasibility of the constructed security model. Further research can be made ranging from enhancing the security power of the EMV specifications to the way that the involved parties communicate with each other. Within this context of current EMV deployments, it is still necessary to protect the confidentiality of cardholder and the integrity of sensitive authentication data, which plays a critical part of for EMV-based transactions and aims to prevent that data being used for fraudulent transactions in other environments. In the near future, it is believed that the EMV should become the means for supporting payments system on the basis of face-to-face channel. Furthermore, according to the Payment Card Industry (PCI)that is a security standards council[47], EMV specifications should also be coupled with a globally adopted robust authentication process for CNP (Card Not Present) transactions, which may help keep the confidentiality of the PAN and sensitive authentication data and reduce transactions fraud.

In conclusion, even though the EMV specifications are public, and obviously very important, there has been surprisingly little public scrutiny of them. Possibly the large size of the specifications, or the fact that they cannot be analyzed without fixing some of the configurations and parameters, have discouraged people. From this point of view, this project provides a brief overview of EMV specifications which has discussed the main cryptographic primitives and security property for it, and also go into details about the transaction protocol phases. Basically, given the large complexity of the EMV specifications and the comprehensive cryptographic techniques, the constructed security model for EMV can be a useful basis for understanding the security requirements of the EMV protocol suite.

As the cryptographic security model for EMV constructed by this project does not include the issuer to agree on a specific transaction, for the future work the generic construction can be extended to include the issuer. In addition, since the analysis of formal syntax and algorithms for the constructed security model are designed based on a pseudo-code, then it is necessary to make this theoretical model executable so that it can improve the practically of the construction and be used to interact with the real EMV-based cards and merchant terminals. Moreover, as a more theoretical approach, one can continue the research from proving the security goals of the constructed model for EMV specifications and to find a more better approach to the problems.

Further to the above, the most interesting area for further improvement and work would be to provide a combination security layer to provide a holistic approach to the objectives of reducing overall fraud and securing cardholder data in the payment industry.

References

[1] Smart Card Alliance. *Card Payments Roadmap in the United States: How Will EMV Impact the Future Payments Infrastructure?* Februry, 2011. Available at: http://www.smartcardalliance.org/resources/pdf/Payments_Roadmap_in_the_US_020111.pdf

[2] Drimer, S., Murdoch S.J., Anderson, R. Failures of Tamper-Proofing in PIN Entry Devices. In *IEEE Security and Privacy*,pages 39-45, November, 2009.

[3] About EMV. Available at: http://www.emvco.com/about_emv.aspx

[4] "2008 Fraud Figures Announced by APACS," In *Assoc. for Payment Clearing Services*, March, 2009.

[5] What is EMV? A TECHNICAL GUIDE TO EMV TRANSACTIONS, COMPLETE WITH A GLOSSARY OF TERMS A FLOWCHART SHOWING THE STAGES OF A TYPICAL TRANSACTION Available at: http://www.emvx.co.uk/emv_guide.aspx

[6] US EMV Business Case. Avaliable at: http://www.gemalto.com/emv/us_bus_case.html

[7] Drimer, S., Murdoch, S.J. and Anderson, R. Optimised to Fail: Card Readers for Online Banking. Cambridge: University of Cambridge Computer Laboratory,UK.

[8] Ruiter, J.D. and Poll, E. Formal Analysis of the EMV Protocol Suite. Digital Security Group Institute for Computing and Information Science (ICIS), Radboud University Nijmegen, 2011.

[9] Balfe, S. Secure Payment Architectures and Other Applications of Trusted Computing. Information Security Group, Royal Holloway, University of London, Egham, Surrey TW2 00EX, UK, 2009.

[10] Khu-Smith, V. and Mitchell, C. Using EMV cards to protect e-commence transactions. Information Security Group, Royal Holloway, University of London, Egham, Surrey, TW20 0EX, United Kingdom.

[11] Mitchell, C. Case Study 1: EMV. Information Security Group, Royal Holloway, University of London, Egham, Surrey TW2 00EX, UK. Available at: http://www.chrismitchell.net/

[12] MasterCard International Incorporated (MasterCard). M/Chip Functional Architecture for Debit and Credit. January, 2006, USA. Available at: http://www.mastercard.com

[13] EMVCo. EMV-Integrated Circuit Card Specifications for Payment Systems, Book 2: *Security and Key Management*, 2008.

[14] EMVCo. EMV-Integrated Circuit Card Specifications for Payment Systems, Book 1: *Application Independent ICC to Terminal Interface Requirements*, 2008.

[15] EMVCo. EMV-Integrated Circuit Card Specifications for Payment Systems, Book 3: *Application Specification*, 2008.

[16] EMVCo. EMV-Integrated Circuit Card Specifications for Payment Systems, Book 4: *Cardholder, Attendant, and Acquirer Interface Requirements*, 2008.

[17] About smart cards: EMV Resources. Available at: http://www.smartcardalliance.org/pages/smart-cards-applications-emv

[18] Herreweghen, V. E. and Wille, U. Risks and Potentials of Using EMV for Internet Payments. In *USENIX Workshop on Smartcard Technology*, Chicago, Illinois, USA, May 1011, 1999.

[19] Visa Smart Debit/Credit Certification Authority Service Description. Version 2.3, June, 2008 Available at: https://partnernetwork.visa.com/vpn/global/retrieve_document.do?...86.

[20] Murdoch, S.J. EMV aws and xes: vulnerabilities in smart card payment systems. Cambridge: University of Cambridge Computer Laboratory, UK, June 16, 2007.

[21] Al-Meaither, M. Secure electronic payments for Islamic finance. Information Security Group, Royal Holloway, University of London, Egham, Surrey, TW20 0EX, United Kingdom,2004.

[22] International Organization for Standardization (ISO), Geneva. ISO 9564-1, *Banking PIN management and security Part 1: PIN protection principles and techniques*, 1991.

[23] Al-Meaither, M. A. and Mitchell, C. J. Extending EMV to support Murabaha transactions. Information Security Group, Royal Holloway, University of London, Egham, Surrey, TW20 0EX, United Kingdom.

[24] Murdoch, S.J., Drimer, S. and Anderson, R. Failures of TamperProofing in PIN Entry Devices. In *IEEE Security & Privacy*, volumn:7, NO.6, pp.39-45, 2009.

[25] Bond, M. Chip & PIN (EMV) interceptor. Cambridge: University of Cambridge Computer Laboratory, UK. Available at: http://www.cl.cam.ac.uk/research/security/banking/interceptor/

[26] Murdoch, S.J., Drimer, S., Anderson, R., and Bond, M. Chip and PIN is Broken. In *IEEE Symposium on Security and Privacy*. Cambridge: University of Cambridge Computer Laboratory, UK, 2010.

[27] Managing fraud with EMV A Risk Manager Checklist for Deploying Chip Technology. In *MasterCard Academy of Risk Management*.

[28] Markantonakisa, K., Tunstallb, M., Hanckea,G.,Askoxylakisc, I. and Mayesa, K. Attacking smart card systems: Theory and practice. In *Information Security Technical Report*, volumn: 14, pp: 45-56, 2009. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.5.5895.

[29] Dent, A. W. Fundamental problems in provable security and cryptography. Information Security Group, Royal Holloway, University of London, Egham, Surrey TW2 00EX, UK.

[30] Pfitzmann, B. and Waidner, M. Composition and integrity preservation of secure reactive systems. In *Proc. 7th ACM Conf. Computer and Communications Security*, Athens, Greece, 1-4 November 2000, pp. 245-254, 2000.

[31] Dent, A. W. and Mitchell, C. J. *User's Guide to Cryptography and Standards*. Artech House, Boston, Massachusetts, USA, 2005.

[32] Nigel, S. *Cryptography: An Introduction*. New York: McGraw-Hill Education, 2003.

[33] Menezes, A., Van Oorschot, P. and Vanstone, S. *Handbook of Applied Cryptography, volume 6 of Discrete Mathematics and its Applications*. CRC Press, Boca Raton, Florida, USA, 1997.

[34] EMVCo, LLC. EMV Card Personalization Specification. Version 1.0, June, 2003. Available at: http://www.emvco.com/specifications.cfm.

[35] Ruiter, J.D. and Poll, E. EMV-the end of skimming?. Digital Security Group Institute for Computing and Information Science (ICIS), Radboud University Nijmegen, 2010.

[36] Bellare, M., Shi, H. and Zhang, C. Foundations of Group Signatures: The Case of Dynamic Groups. In *Topics in Cryptology CT-RSA 2005Lecture Notes in Computer Science*, Volume 3376/2005, pp.136-153, 2005.

[37] Goldwasser, S., Micali, S. and Rivest, R. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal of Computing*, 17(2):281-308, April 1988.

[38] Rackoff, C. and Simon, D.Non-interactive zero-knowledge proof of knowledge and chosen-ciphertext attack.In *Advances in Cryptology - Crypto '91*, LNCS Vol. 576, J. Feigenbaum ed., Springer-Verlag, 1991.

[39] Bellare,M., Desai, A., Jokipii, E. and Rogaway, P.A concrete security treatment of symmetric encryption, In *Proceedings of the 38th Symposium on Foundations of Computer Science*, IEEE, 1997.

[40] Desa, A. New Paradigms for Constructing Symmetric Encryption Schemes Secure Against Chosen-Ciphertext Attack. Department of Computer Science & Engineering, University of California at San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA.

[41] Sahai,A. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS'99*, pages 543-553, 1999.

[42] Preneel, B. and van Oorschot, P.C. On the Security of Iterated Message Authentication Codes. *IEEE Transactions on Information Theory*, 45(1):188-199, 1999.

[43] Paul, D.A., Jeremy, D.E, Ken, B.and Michael, P. Authenticated Encryption: Combining Authentication with Encryption to get IND-CCA. September 21, 2004. Available at: www.cs.cmu.edu/ hopper/crypto_course/lecture9.pdf

[44] Bellare, M. and Namprempre, C. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. DBSEM, May 11, 2004. Available at:publications.nr.no/AuthenticatedEncryption.pdf

[45] Groth, J.Simulation-Sound NIZK Proofs for a Practical Language and Constant Size Group Signatures.UCLA, Computer Science Department, 3531A Boelter Hall,Los Angeles, CA 90095, USA,December 10, 2006.

[46] Feige, U., Lapidot, D. and Shamir, A. Multiple non-interactive zero-knowledge proofs under general assumptions. *SIAM Journal on Computing,29(1):1-28, September, 1999.*

[47] Payment Card Industry. PCI DSS Applicability in an EMV Environment A Guidance Document.Version 1,October,2010. Available at: https://www.pcisecuritystandards.org/security_standards/

Bibliography

1 Bellare, M.. Pointcheval,D. and Rogaway, P. (2000) Authenticated Key Exchange Secure Against Dictionary Attack. *Advances in Cryptology - Eurocrypt 2000 Proceedings, Lecture Notes in Computer Science* Vol. 2656, E. Biham ed, Springer-Verlag.

2 Fiore,D. Gennaro, R. and Smart, N.P. (2010) Constructing Certificateless Encryption and ID-Based Encryption from ID-Based Key Agreement. *Pairing-Based Cryptography - Pairing 2010*, pp. 167186. November 2010.

3 Bellare, M., Namprempre, C. and Neven, G. Security Proofs for Identity-Based. in *Advances in Cryptology-EUROCRYPT*, volume 3027 of Lecture Notes in Computer Science, C. Cachin and J. Camenisch ed.,Springer-Verlag, 2004.

4 Fiat, A. and Shamir, A. How to prove yourself: Practical solutions to identication and signature problems. In A. Odlyzko, editor, *CRYPTO 1986 ,volume 263 of LNCS* ,pages 186-194. Springer-Verlag, August 1986. (Cited on page 3, 5, 6, 20, 24.)

5 Cha,J.C. and Cheon, J.H. An identity-based signature from gap diehellman groups. In Y. Desmedt, editor,*PKC 2003, volume 2567 of LNCS* pages 18-30. Springer-Verlag, January 2003. (Cited on page 3, 4, 5, 6,12, 33, 34.)

6 Tsiotras, C. *Security evaluation of on-line banking systems.* Department of Computer Science, Univesity of Bristol, 2008.

7 Anderson, R. J. *Security Engineering: A Guide to Building Dependable Distributed Systems, Second Edition.* Wiley Publishing Inc, Indianapolis, Indiana, USA, 2008.

8 Song, S.H., Lee, J.H. and Ryou, J.C. Payment PKI based on EMV and Efficient IC Card Authentication Mechanism. Chungnam National University, Korea, October, 2004.