Contactless Payments:
Usability at the Cost of Security?

Martin J. Emms

School of Computing Science, Newcastle University

Newcastle Upon Tyne, UK

April 2016

This thesis is submitted for the degree of

Doctor of Philosophy

ABSTRACT

EMV (Europay, MasterCard, Visa), commonly termed "Chip & PIN", is becoming the dominant card based payment technology globally. The EMV Chip & PIN transaction protocol was originally designed to operate in an environment where the card was physically inserted into the POS terminal / ATM and used a wired connection to communicate. The introduction of EMV contactless payments technology raises an interesting question *"has usability been improved at the cost of security?"*. Specifically, to make contactless payments more convenient / usable, a wireless interface has been added to EMV cards and PIN entry has been waived for contactless payments. Do these new usability features make contactless cards less secure?

This PhD thesis presents an analysis of the security of the EMV contactless payments. It considers the security of the EMV contactless transaction protocols as stand-alone processes and the wider impact of contactless technology upon the security of the EMV card payment system as a whole.

The thesis contributes a structured analysis methodology which identifies vulnerabilities in the EMV protocol and demonstrates the impact of these vulnerabilities on the EMV payment system. The analysis methodology comprises UML diagrams and reference tables which describe the EMV protocol sequences, a protocol emulator which implements the protocol, a Z abstract model of the protocol and practical demonstrations of the research results. Detailed referencing of the EMV specifications provide a documented link between the exploitable vulnerabilities observed in real EMV cards and the source of the vulnerability in the EMV specifications.

Our analysis methodology has identified two previously undocumented vulnerabilities in the EMV contactless transaction protocol. The potential existence of these vulnerabilities was identified using the Z abstract model with the protocol emulator providing experimental confirmation of the potential for real-world exploitation of the vulnerabilities and test results quantifying the extent of the impact.

Once a vulnerability has been shown to be exploitable using the protocol emulator, we use practical demonstrations to show that these vulnerabilities can be exploited in the real-world using off-the-shelf equipment. This presents a stronger impact message when presenting our research results to a non-technical audience. This has helped to raise awareness of security issues relating to EMV contactless cards, with our work appearing in the media, radio and TV.

Acknowledgements

Professor Aad van Moorsel my PhD Supervisor and great support, he has believed in me and in this PhD project.

Robert Emms and Lynne Marshall-Dunn for their endless patience in many long hours of proof reading and copy editing in this thesis. Heartfelt thanks for all of the support and encouragement over the past months whilst I wrote this PhD thesis.

Dr Budi Arief my co-author on several conference papers including FC 2013 Contactless Verify PIN and CCS 2014 Foreign Currency Flaw. Budi's guidance in the art of writing papers has been invaluable and he is above all a good friend.

Dr Leo Freitas with whom I developed the analysis methodology described in Chapter 5 and is co-author on the CCS 2014 Foreign Currency Flaw.

Joseph Hannon who developed the Android software for several of the practical demonstrations described in this thesis including the practical demonstration for CCS 2014 Foreign Currency Flaw. Joe also developed the Android card emulator code. Joe was an excellent student and is a good friend. Joe has gone on to a fantastic start to his career.

Nicholas Little who developed the multiple card reader described in FC 2013 Contactless Verify PIN. From meeting Nick as a 1st year student it was obvious that he was an exceptional programmer.

Troy Defty was a confident and accomplished student who helped develop some of the Android practical demonstrations including the contactless relay.

Mike Bond and Ross Anderson for encouraging the use of practical experiments in my research, to demonstrate that the vulnerabilities we have identified in the protocol were exploitable in the real world. Thereby increasing the impact of my research.

Table of Contents

List of Tables

List of Figures

Glossary of Terms

3-DES	Triple DES – is the common name for Triple Data Encryption Algorithm which applies the Data Encryption Standard (DES) three times to each data block. 3-DES is a 64 bit block-cipher
AAC	Application Authentication Cryptogram – is the application cryptogram generated by an EMV card indicating that the transaction has been cancelled.
AC	Application Cryptogram – the EMV protocol utilises 3-DES encoded cryptograms as a secure method of communication between the EMV payment card and the Issuing Bank. There are four different types of cryptogram; TC (transaction approved), AAC (transaction declined), ARPC (request online approval from the Issuer) an ARPC (Issuer authorisation decision).
Acquirer	Refers to the bank that holds the destination bank account for the transaction, which is typically the bank that issued the POS terminal to the merchant. Also referred to as the "acquiring bank".
AFL	Application File Locator – a list of records on the card that contains all of the data required by the POS terminal to complete the transaction. Included in this data are the RSA keys required for the POS terminal to validate the transaction.
Aic	Acquirer implicit certificate – from our work on POS Authentication Chapter 8, the Aic is an ECQV implicit certificate used to generate the acquirer public key during the POS authentication process.
Aid	Acquirer identification data – from Chapter 8, the Aic is the data used to uniquely identify the acquirer, which is encoded into the acquirer's ECQV implicit certificate (Tic), consisting of the acquirer ID, expiry date and a sequence number.
APDU	Application Protocol Data Unit – the APDU is the messaging structure used by ISO-7816 compliant smartcards such as EMV payment cards.
Apk	Acquirer public key – from Chapter 8, the Aic is generated by the card from the Aic.
ARPC	Authorisation Response Cryptogram – in the protocol sequence for transactions that require online authorisation, the card generates an Authorisation Request Cryptogram

	to signify that it wishes to complete the transaction online. The Issuer responds with and ARPC, which encodes the Issuers authorisation response to the transaction request. The ARPC is 3-DES or AES encrypted using the Issuers private key which allows the card to ensure that the card to validate that the response came from the Issuer and has not been altered.
ARQC	Authorization Request Cryptogram – is generated by the card to indicate that it wants to complete the transaction online (see ARPC).
Ask	Acquirer private key – from Chapter 8, the Ask is generated by the Certificate Authority and stored by the acquirer.
ATM	Automatic Teller Machine – commonly termed cash machines, they allow bank customers to withdraw cash from their bank account.
CA	Certificate Authority – in EMV each of the card scheme providers Visa, MasterCard, American Express, JCB, Discover and Union Pay act as the Certificate authority for their own branded cards. The CAs generate the Issuer's RSA private keys thereby allowing the Issuers to generate RSA public private key pairs for their cards. ATMs and POS terminals can validate the cards RSA signature using the CAs public key.
CAind	Certificate Authority public key index – from Chapter 8, an EMV POS terminal supports multiple CA public keys. The card indicates the CA public key that it supports using the CA public key index.
CApk	Certificate Authority public key – from Chapter 8, the POS terminals and ATMs use the CA public key as the know starting point to validate the Issuer public key, the card public key and the digital signature.
Cardholder	This is a generic term for the person with an EMV credit or debit card who is making a payment in a shop / restaurant or is withdrawing cash from an ATM. The term also implies that the person has a bank account to which the card is attached.
CAsk	Certificate Authority private key – from Chapter 8, the CA private key is used to generate the Issuer public private key pairs.
Chip & PIN	The common name used in the UK to refer to EMV payment system. The EMV specifications defines the operation of the payment cards, POS terminals and ATMs.
ECC	Elliptic Curve Cryptography – is a public key cryptography system. It is termed ECC because the cryptography algorithm utilises the discrete points along an elliptic curve, described by the equation $y^2 = x^3 + ax + b$.

ECDSA	Elliptic Curve Digital Signature Algorithm – a method of generating digital signatures based on the elliptic curve cryptography system.
ECQV	Elliptic Curve Qu-Vanstone
EMV	Europay MasterCard Visa is a global standard for card payments (commonly termed "Chip & PIN"). The standard ensures the interoperability of EMV payment cards, ATMs and POS terminals across different banks and in different countries.
EMVCo	Is the organisation that controls the EMV specifications. It is a collaboration between the card scheme providers American Express, Discover, JCB, MasterCard, UnionPay and Visa. It also includes associate members (government bodies, banks, payment providers, retailers and utilities) who add to the technical and operational issues.
ISO-14443	International standard governing contactless (proximity) smartcards. ISO-14443 the operation of describes ISO-7816 smartcards with a contactless interface.
ISO-7816	International standard governing the physical characteristics, operation and messaging protocol of integrated circuit smartcards.
Issuer	The bank that issued the card used in the transaction and holds the source bank account for the transaction.
ITSO card	NFC enabled travel card.
NFC	Near Field Communication
nonce	A cryptographic nonce (*Number ONCE*) is a term given to random or pseudo random number which changes each time an authentication protocol is performed. The nonce is used in cryptography to protect cryptographic hash functions and digital signatures against the known plaintext attacks. In EMV the nonce is an unpredictable number which is used as the challenge sent by the card to the POS terminal. The POS terminal digitally signs the unpredictable number thereby preventing replay attacks.
Oyster card	NFC enabled travel card.
PAN	Primary Account Number; the PAN is the universal ISO 7812 compliant 16 card number which identifies each individual payment card. The PAN is printed on the front of the card and is encoded into the smartcard interface data which is transmitted to the POS terminal for payment.
Payment card	This is a generic term for credit and debit cards. In this thesis credit and debit cards are considered to be operationally the same as they perform the same transaction protocol and are subject to the same card payment scheme clearing processes.

PDOL	Processing options Data Object List – A flexible list of data fields requested by the card. The POS returns the requested data fields in the GetProcessingOptions() message.
PKI	Public Key Infrastructure, is a mechanism by which a trusted 3rd party (the Certificate Authority) can validate the identity of the parties in a cryptographic exchange.
POS	Point of Sale terminal; the device used in shops and restaurants to make card payments. A POS terminal may also describe an unattended vending machine such as a parking meter with a card payment interface.
RSA	Is an asymmetric public key cryptosystem named after its three co-inventors Ron Rivest, Adi Shamir, and Leonard Adleman. RSA is used to encrypt / decrypt messages and to produce digital signature. EMV credit and debit cards use RSA to produce digital signatures which can be used to validate that the card is genuine.
SAM	Secure Access Module – Tamper proof storage module used by POS terminals for cryptographic key storage.
SDAD	Signed Dynamic Authentication Data – In the current EMV system the card authorised the transaction data by signing the transaction data with its private key to produce the SDAD.
SFI	Short File Indicator – The storage on EMV cards is in blocks of 16 records, each block is referenced by a unique SFI.
TC	Transaction Certificate – Response code from the card that indicates that the transaction has been successful.
Tic	Terminal implicit certificate - ECQV implicit certificate used to generate the terminal public key during the POS authentication process.
Tid	Terminal identification data – Data to uniquely identify the terminal, which is encoded into the terminal's ECQV implicit certificate (Tic), consisting of the terminal ID, expiry date and a sequence number.
Tpk	Terminal public key - Generated by the acquirer and stored on the POS terminal's Secure Access Module.
Tsk	Terminal private key - Generated by the acquirer and stored on the POS terminal's Secure Access Module.
UPN	UnPredictable Number – This is a nonce generated by the card and included in the SDAD signed transaction data.

Chapter 1. Introduction

Card payments have become ubiquitous, we pay for our everyday items in store, we make online purchases and we can even transfer money to our friends and family, all using our credit / debit cards. EMV is the smartcard based payment technology chosen as the global electronic card payment technology. EMV has already replaced magnetic stripe in 76 countries, including most of Europe, and is currently being adopted in the United States of America, China and India [1] [2] [3]. This will make it the dominant card payment technology globally

EMV smartcard technology introduces significant security improvements over the magnetic stripe payment cards that they replace. It has made the cards difficult to copy, but, because they also need to deal with the complexities of the technologies they are superseding, that security can be put at risk. From our research it is contended that contactless payments technology adds to the security issues of the EMV protocol.

In 2008 EMV contactless payments were introduced into the UK, they are a more convenient method of payment for purchases under £20 (May 2015). This convenience is achieved by fundamentally changing the way in which EMV payment cards are used. Where previously the cardholder needed to insert the card into a POS terminal and enter their PIN to make a transaction, now contactless payments can be made wirelessly by tapping the card on the POS terminal and the cardholders PIN is not required.

1.1 Aims and Objectives

This thesis aims to answer the question *"has usability been improved at the cost of security?"*. Specifically do the features which make contactless payments more convenient and user friendly impact the security of the EMV payment system:

- does the wireless interface enable new categories of wireless attack that were not previously possible with the Chip & PIN cards?
- what is the scope of the impact of the removal of the PIN verification step, are there any unforeseen vulnerabilities created?
- contactless introduces seven new protocol sequences, with varying authentication processes, how does this increased complexity impact security?

To answer these questions, I have performed a structured analysis of the EMV contactless transaction protocol to identify any potential vulnerabilities. I have also created a functioning emulation of the EMV contactless protocol, which demonstrates the practical security impacts of the introduction of contactless technology upon the existing EMV payments system.

1.2 Thesis Contributions

The work carried out as part of this PhD research has contributed to the subject of payments security research in the following ways:

- Development of a structured analysis methodology for EMV contactless transaction protocols which identifies flaws, demonstrates they exist in the real-world and links the vulnerability to its origin in the EMV specifications.
- Implementation of a protocol emulator which can execute protocol scenarios which target specific contactless protocol vulnerabilities and thereby allow us to quantify the resulting security impact.
- We have published research identifying two previously undocumented vulnerabilities in the EMV contactless transaction protocol.
- A literature review which analyses the wider impacts of contactless payments upon the security impact of the EMV payment system, outside the protocol analysis presented in this PhD thesis.

1.3 Publications

This thesis presents a body of PhD research which incorporates contributions which have been previously published as conference papers and as technical reports:

- Emms M, Arief B, Freitas L, Hannon J, and van Moorsel A, "Harvesting High Value Foreign Currency Transactions from EMV Contactless Credit Cards without the PIN," in *21st Conference on Computer and Communications Security (CCS 2014)*, Phoenix AZ, 2014.
- Emms, M., Arief, B., Little, N.J., Van Moorsel, A, "Risks of Offline Verify PIN on Contactless Cards.," in *17th International Conference on Financial Cryptography and Data Security*, Okinawa Japan, 2013.
- Freitas, L. and Emms, M, "Formal specification of EMV protocol. (TR 1429*)," "* Newcastle University - School of Computing Science Technical Report Series, 2014.
- Emms M, Freitas L and van Moorsel A,, "Rigorous Design and Implementation of an Emulator for EMV Contactless Payments (TR 1426)," Newcastle University - School of Computing Science Technical Report Series, 2013.

- Emms M, Arief B, Hannon J, van Moorsel A. "POS Terminal Authentication Protocol to Protect EMV Contactless Payment Cards (TR 1386)". Newcastle University - School of Computing Science Technical Report Series, 2013.
- Emms, M., "Impacts of Data Leakage from Contactless Payments Cards" a lightning talk at *Symposium On Usable Privacy and Security (SOUPS 2013)*, Newcastle UK, 2013
- M. Emms, "Practical Attack on Contactless Payment Card," in *Human Computer Interaction (HCI 2011) Workshop - Health, Wealth and Identity Theft*, Newcastle UK, 2011.

The security of contactless payments affects everyone who carries a contactless card in their wallet. Our practical attack scenario implementations help us to convey a security message in a way that is accessible to the general public. We have used the media to carry this message to a wider audience warning of potential security vulnerabilities.

Our work has also featured in media reports on TV, Radio and in news articles. This helps to communicate our security research to the general public who, as EMV cardholders, are potentially impacted by the vulnerabilities identified.

- BBC News 24, "BBC Tyne - Visa card glitch could lead to fraud," 01 November 2014.
- BBC Radio 4, "Money Box - Card payment glitches at contactless terminals," 19 May 2013.
- Sky News, "Contactless Cards: App Reveals Security Risk," 04 June 2013.
- BBC Radio 4, "Today," 03 June 2013
- BBC Radio 5 Live, "Saturday Edition," 18 May 2013
- BBC Radio 4, "You & Yours - Portas towns and contactless card payments," 30 May 2013.
- BBC Radio 2, "Drivetime," 30 May 2013.
- Info Security Magazine, "UK customers charged twice with contactless payment cards," 20 May 2013
- Mail Online, "How 30million 'wi-fi' credit cards can be plundered by cyber identity thieves exploiting contactless payment technology," 01 June 2013
- The Week, "Contactless debit and credit cards: what are the risks?," 30 May 2013
- The Times, "Amazon customers at risk over contactless payments," 04 June 2015.
- gulli.com, "NFC: Irrtümliche Zahlungen erfolgt," 22 May 2013.

1.4 Structure of Thesis

Chapter 2 describes the global payment system and each of the card payment methods. It describes contactless payments and where they fit into the global payment system as a whole. This chapter briefly examines the relationship between the different card payment technologies and card fraud committed in the UK.

Chapter 3 describes the EMV transaction protocol, setting it in the context of the EMV card payment system. This chapter provides a comparative analysis of the different transaction authorisation methods used for contact "Chip & PIN" payments and contactless payments, highlighting the strength and weaknesses within each protocol.

Chapter 4 looks at the published research relating to contactless payments. The focus being on (i) analysis of transaction protocols (ii) exploitable vulnerabilities in transaction protocols (iii) exploitable vulnerabilities in the contactless technology.

Chapter 5 describes the analysis methodology developed for this PhD research. The methodology identifies flaws in the contactless transaction protocol and, by means of a protocol emulator, demonstrates the existence of the flaws as exploitable vulnerabilities in EMV cards and/or POS terminals.

Chapter 6 describes the foreign currency flaw in the contactless transaction protocol which was identified using the analysis methodology described in Chapter 5. Our research found that all EMV cards (Chip & PIN and contactless) have the foreign currency flaw. However, it is the usability features, i.e. the wireless interface and no PIN that makes this flaw exploitable on contactless cards.

Chapter 7 describes the contactless verify PIN protocol flaw, which again was identified using our analysis methodology. This chapter demonstrates that the wireless interface directly impacts on one of the existing security features of EMV Chip & PIN cards

Chapter 8 describes a potential solution for the vulnerabilities identified in the contactless transaction protocol by confirming that the transaction device is authentic. The solution aims to prevent unauthorised access to the contactless payment card which is at the root of many of the vulnerabilities identified by our research.

Chapter 9 describes the nature of the research, involving real EMV contactless cards and terminals, and favoured a practical approach in developing our software and analysis tools. This chapter contains groundwork experiments required for this PhD thesis.

Chapter 10 contains the conclusion to my PhD research. It draws upon the results of my PhD research to evaluate the question "Contactless payments: usability at the cost of security?".

Chapter 2. Context of Electronic Payments

This chapter gives an overview of the EMV (Europay, MasterCard, Visa) payment system, to give the reader an understanding of the global payments environment in which contactless payment cards exist. The thesis focuses on EMV payment transactions (i.e. card purchases), this section, therefore, does not cover the details of other EMV transaction types; chargeback (refunds), referral (telephone payments) and pre-authorisation (hotels and rental cars).

It will also discuss some of the latest developments in payments technologies as applicable to the research presented in this thesis. First, it will introduce several terms used throughout this thesis which refer to the different parties involved in an EMV card payment.

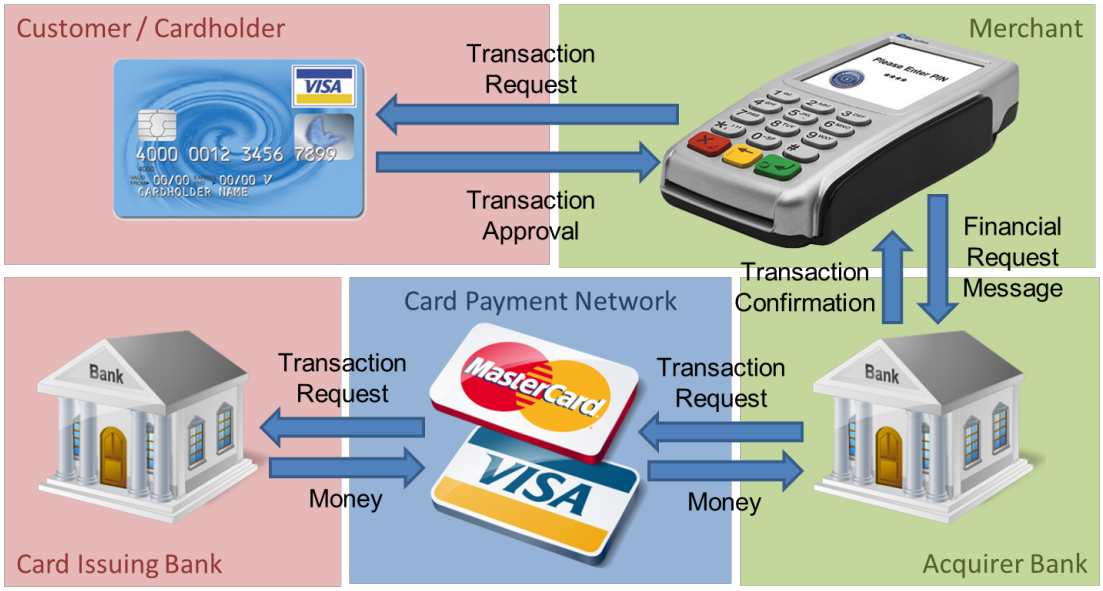


Figure 1 - Card Payment Operation

Figure 1 illustrates the card payment process, it shows the relationship between the entities involved in a payment and the information flow between the entities.

- Customer / Cardholder, predominantly referred to as the cardholder in this thesis
- Issuing Bank is the customer's bank
- Merchant is a shop or restaurant with a POS terminal,
- Acquirer Bank, is the merchant's bank
- Card Payment Network is operated by the card payment schemes; MasterCard, Visa, American Express, JCB, Discover and UnionPay.

Customer / Cardholder: The cardholder is the person making a card payment in a shop or restaurant. The cardholder has a payment card issued to them by the bank which holds their account. The customer uses their card at the POS terminal make a payment.

Merchant: refers to a shop, restaurant or business which has a POS terminal to receive payments from its customers. The merchant must have a merchant account at the acquirer bank. In the UK, merchant accounts were traditionally subject to more rigorous checks than customer / cardholder accounts. However, this has changed with the introduction of iZettle and PayPal mobile payment terminals (see section 2.6.2).

POS Terminal: communicates with the customers EMV payment card to request a payment to the Merchant. The card can authorise the payment immediately, in offline mode, or request additional online authorisation from the Issuing bank. In online mode the POS terminal relays messages between the card and the Issuing Bank, which facilitate online authorisation of the transaction.

Acquirer Bank: provides banking services to the merchant, including collection of payments from the POS terminal. The acquirer bank performs initial checks to verify that the card is not lost or stolen. The transaction is then routed to the appropriate Issuing Bank using the Primary Account Number (PAN), which is the 16 digit number printed on the front of the card, and the service code from the card data.

The Acquirer bank is responsible for formatting the transaction data and sending it to the payments network which will forward it on to the Issuing Bank. In the formatting process the acquirer bank will add many data fields which are captured by the POS terminal and are required by the Issuing Bank to complete the transaction.

Card Payment Network: is the infrastructure which routes transaction requests from merchant, via the acquirer bank, to the issuer bank and returns the transaction approval / decline response from the issuer. The EMV card payment network consists of several networks run by the card scheme providers, Visa, MasterCard, American Express, JCB, Discover and Union Pay. Transactions for each card type are routed through the network of the associated card scheme provider, for example all Visa card transactions are routed through the Visa Network.

Card Issuing Bank: The issuing bank holds the customer's account and issues them with a payment card. When the issuer receives a transaction request:

- the cardholder's account is checked to see if there are sufficient funds
- the cardholder's account balance is debited by the transaction amount
- a payment authorisation is forwarded through the card payment clearing system to the acquirer bank

The card Issuing Bank must deal with any disputes from the customer and is responsible for fraud losses where neither the cardholder nor the merchant is deemed to be at fault.

2.1 EMV Smartcard Payment Technology

EMV is a global specification for smartcard based payments, commonly termed "Chip & PIN". EMV represents a step forward in security from the magnetic stripe technology it replaces. The improved security is based on the "Chip" which is capable of generating a cryptographic signature to validate transactions. It also relies on the "PIN" which is used by the cardholder to authorise transactions and is harder to forge than the signature authorisation previously used with magnetic stripe cards.

There are two distinct payment technologies covered by the EMV standards, contact Chip & PIN payments and contactless payments. EMV is primarily an interoperability standard, it aims to ensure that all EMV payment cards are compatible with all ATMs and all POS terminals worldwide. The objective is that the customer should always be able to use their card anywhere in the world, which is very convenient for the customer. It also maximises the transaction fees generated for the acquirer and Issuing Banks.

In addition to interoperability the EMV standards introduced significant security improvements over the magnetic stripe technology that preceded it. As the Chip & PIN moniker suggests, EMV utilises a smartcard chip technology that provides significant security improvements over magnetic stripe cards. Key to these security improvements is the smartcard's ability to dynamically authorise transactions and provide proof of authorisation in the form of a cryptographic signature. Magnetic stripe cards authorise transactions using static authorisation data [4], which can be easily copied as described in section 2.5.1.

EMV Standards: The EMV standard defines the protocols, technology and information standards that the payment cards, ATMs and POS terminals must comply with to guarantee interoperability among various cards, POS terminals and ATMs. The EMV standards also cover interoperability with magnetic stripe cards, ATMs and POS terminals.

Global Deployment of EMV Technology: Figures released by EMVCo in May 2013 [5] report that globally there are 1.62 billion EMV payment cards in circulation and 23.8 million terminals (POS terminals, ATMs and vending machines). The purpose of the EMV standards [6] [7] is to ensure that EMV payment cards work everywhere that the cardholder may wish to use their card. The EMV standards also ensure that EMV payment cards and terminals are interoperable with magnetic stripe cards and terminals. Magnetic stripe is the card payment standard which preceded EMV and it is still in operation in many countries such as the USA and China (at time of writing, May 2015).

The EMV protocol standards must therefore incorporate competing (and sometimes conflicting) requirements from the card scheme providers, MasterCard, Visa, Amex, JCB, Diners, Discover,

UnionPay, the card Issuing Banks, the merchant acquirer banks as well as the financial regulators in each of the countries in which EMV operates. Consequently, the EMV protocol standards are complex and contain many features to facilitate backward compatibility.

Magnetic Stripe Compatibility: The EMV smartcard payment system is being adopted worldwide, replacing magnetic stripe. This process has been a gradual migration with both payment systems supported in parallel. Even in countries like the UK where there is 100% adoption of EMV cards, ATMs and POS terminals magnetic stripe must be supported (i) to allow visitors from other countries with magnetic stripe cards to use UK ATMs and POS terminals (ii) to allow UK cardholders to use EMV cards whilst visiting countries that use magnetic stripe technology.

Backward Compatibility Legacy Data: The payment network that supports the new EMV payments technologies (Chip & PIN, contactless and mobile phone payments) is essentially the same payment network that supported the magnetic stripe payment cards. As such, for a transaction to be successfully processed by the payment network EMV retains many data structures from magnetic stripe (e.g. Track 2 equivalent data, service code and PVV) which would otherwise be redundant in a purely EMV smartcard environment.

2.2 EMV Payment Cards (smartcards)

The design imperative of the EMV payment cards is usability, to make it as convenient as possible for the cardholder to make the payment in all possible situations. This is achieved by using the four payment methods identified in Figure 2; (1) Chip & PIN payments, identified here by the contact chip present on the card (2) contactless payments, cards with contactless payment capability display this logo (3) magnetic stripe payments, EMV cards have a magnetic stripe on the rear of the card for backward compatibility with countries, such as the USA, where EMV has not been adopted (4) online payments and telephone payments (termed card-not-present payments), are made using the card data printed on the front of the card and the CVV2 printed on the back.

Figure 2 - EMV Card Payment Methods

(1) Chip & PIN payments: the card is inserted into a POS terminal or ATM and communication with the card is established via the exposed contacts on the surface of the card. In countries where EMV has been adopted, Chip & PIN is the preferred method of payment in shops, restaurants and at ATMs.

When Chip & PIN was introduced it brought two significant security improvements. The smartcard chip was harder to clone than the existing magnetic stripe cards. The PIN made it harder for criminals to use lost / stolen cards where previously a signature could be easily forged.

(2) Contactless payments: utilises a wireless connection (NFC) to establish communication with the card. Contactless payments are intended to be a faster and more convenient way to pay for small purchases than the existing Chip & PIN, whilst still retaining many of the security features. Contactless transactions share the same chip technology on the card and have a very similar protocol sequence to the Chip & PIN transactions. The main difference is that the cardholder is no-longer required to input their PIN to authorise the transaction.

(3) Magnetic stripe payments: the POS terminal / ATM reads the static card information from the magnetic stripe on the back of the card. Magnetic stripe was the first electronic payment technology and is still (May 2015) the most widely used payment technology worldwide. The information contained in the magnetic stripe includes the 16 digit card number, cardholder name, card expiry date, the PIN Verification Value (PVV) and the routing information for the payments network. This data is in a specific format detailed in ISO-7813 [4]

(4) Card-not-present payments: the information printed on the card (i.e. 16 digit card number, customer name, card expiry date and card CVV2) are used to make online payments and telephone payments. The information is printed / embossed in accordance with ISO 7811 [8]

2.3 EMV Smartcard Security Features

EMV is based on smartcard technology, which has the processing power to support much more complex security features than the previous magnetic stripe cards. The essential difference between an EMV smartcard and a magnetic stripe card is that the chip on the smartcard allows the EMV card to actively participate in the transaction process, by generating a non-repudiable cryptographic transaction approval. This is not true of magnetic stripe cards where the POS terminals and ATMs simply use the static data contained in the magnetic stripe to approve the transaction.

EMV cards are considered to be harder to clone than the magnetic stripe cards they replaced, because:

- EMV smart cards cryptographically approve transactions by generating an Application Cryptogram. The cryptogram validates the transaction data and proves that the card which authorised the transaction is a genuine card.
- EMV smartcards provide secure, "tamper proof", storage for the card's private cryptographic keys and for the card's PIN. This means that although the data can be skimmed from an EMV smartcard, the private keys required to authorise a transaction cannot be copied, thereby preventing the EMV card from being cloned.

- The technology required to skim the data from the smartcard chip on Chip & PIN cards Figure 3 is much more sophisticated than magnetic stripe skimmers Figure 7
- Magnetic stripe skimming devices Figure 7 would typically be placed externally on an ATM, Chip & PIN skimmers Figure 3 need to be placed inside the POS terminal which are now being made tamper proof to stop them being opened.
- In the early years of EMV the programmable smartcards required to create the clones were expensive and hard to obtain.
- Programming the smartcard functionality is more technically challenging than simply copying the data in magnetic stripe.

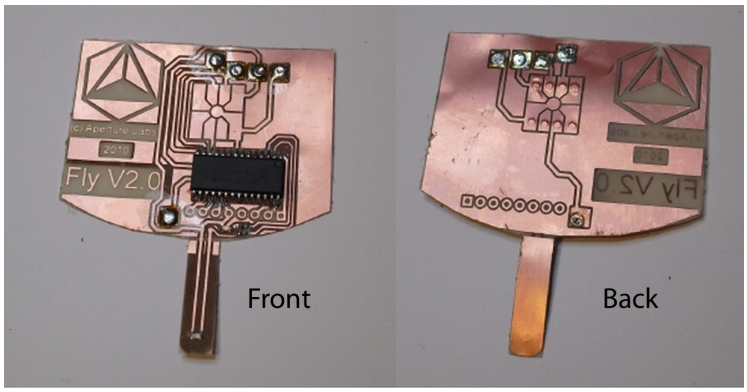


Figure 3 - Chip & PIN Skimming Device [9]

Figure 3 show the front and rear of the Chip & PIN skimming device. What is notable about the Chip & PIN skimmer is the complexity of the circuitry and the fact that the circuit must be wafer thin to fit in the gap between the card and the electrical contacts in the POS terminal.

2.4 Evolution of Card Payments

The first universally accepted credit cards appeared in the 1950s, prior to which the larger merchants each had their own card payment scheme and the cards were not accepted anywhere else. During the 1960s and 1970s the number of outlets accepting card payments and the number of customers carrying credit cards steadily grew.

Figure 4 - Card Imprinter

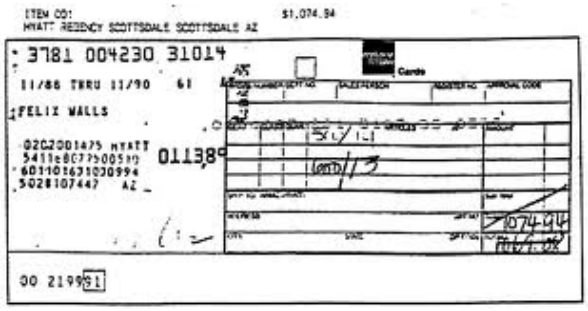


Figure 5 - Card Imprint Slip

Card payments were originally a paper-based transaction; a carbon paper impression of the card was taken using a card imprinter, Figure 4. The raised printing (embossing) [8] on today's credit and debit

cards (customer name, 16 digit card number and expiry date) allows transactions to be recorded by card imprinting. The imprinter presses the carbon paper down onto the raised printing on the card leaving a copy of the card details behind.

To complete the transaction the merchant also fills in the transaction details (item description, amount and date) and asks the customer to sign the card imprint slip, Figure 5. The slips are posted to the bank for processing and the money would follow some days to a week later.

In the 1980s and 1990s global electronic payment networks were developed to support the introduction of ATMs and electronic POS terminals. Electronic payment gradually replaced the paper-based system, bringing benefits to customers, merchants and payments providers.

2.4.1 Electronic Payments

When electronic payments were introduced they used magnetic stripe technology to allow payments cards to be read electronically by ATMs and by POS terminals.

Electronic payments had the following major advantages over the paper-based payments:

Cheaper: It cost the banks less to process each payment. This in turn benefitted merchants who were charged less commission by the banks. Finally it benefitted customers because more merchants were willing to accept card payments and in the case of some merchants who had a surcharge for card payments this was dropped.

Faster: Merchants received their payments in minutes / hours rather than days / weeks. This again encouraged more merchants to adopt card payments, thereby benefitting both customers and banks.

More secure: In an online environment every transaction could be automatically checked for lost / stolen cards and accounts with insufficient funds. This could be done in real-time whilst the customer was still in store. Previously this could only be done by making a telephone call to the Issuing Bank, which was becoming increasingly impractical as the volume of credit card transactions increased.

2.4.2 Magnetic Stripe Technology

Magnetic stripe was the first electronic payment technology. It is also the simplest. The magnetic stripe on the payment card contains two static data blocks “Track 1” and “Track 2”. The two tracks store all of the data elements required (in a prescribed format) to authorise the transaction and route it to the appropriate Issuing Bank for payment [4].

2.5 Card Fraud Overview

In this section I will give an overview of card fraud as it affects the global payments system. Figure 6 shows UK card fraud statistics from 1995 to 2014 [10] [11]. The statistics reveal that ratio between the different types of card fraud changes year on year. In Figure 6 the blue line represents face-to-face forms of card fraud, where stolen and counterfeit cards were used in shops and at ATMs. The

red lines represent remote (card-not-present) forms of card fraud, where stolen card details were used to purchase goods over the telephone and online. From the statistics it can be seen that the type of fraud has changed significantly since 1995. In 1995 face-to-face fraud was the predominant form of fraud representing 92.3% the total UK card fraud and card-not-present fraud represented just 5.5%. In 2014 card-not-present fraud has increased to be the most common from of fraud representing 69.2% of total UK card fraud and face-to-face fraud has reduced to just 24.6%. The green line depicts one particular type of card fraud, fraud committed abroad. These statistics have been included as they illustrate a sharp increase in overseas card fraud in 2007 and the Issuing banks' response to stop this particular type of fraud.

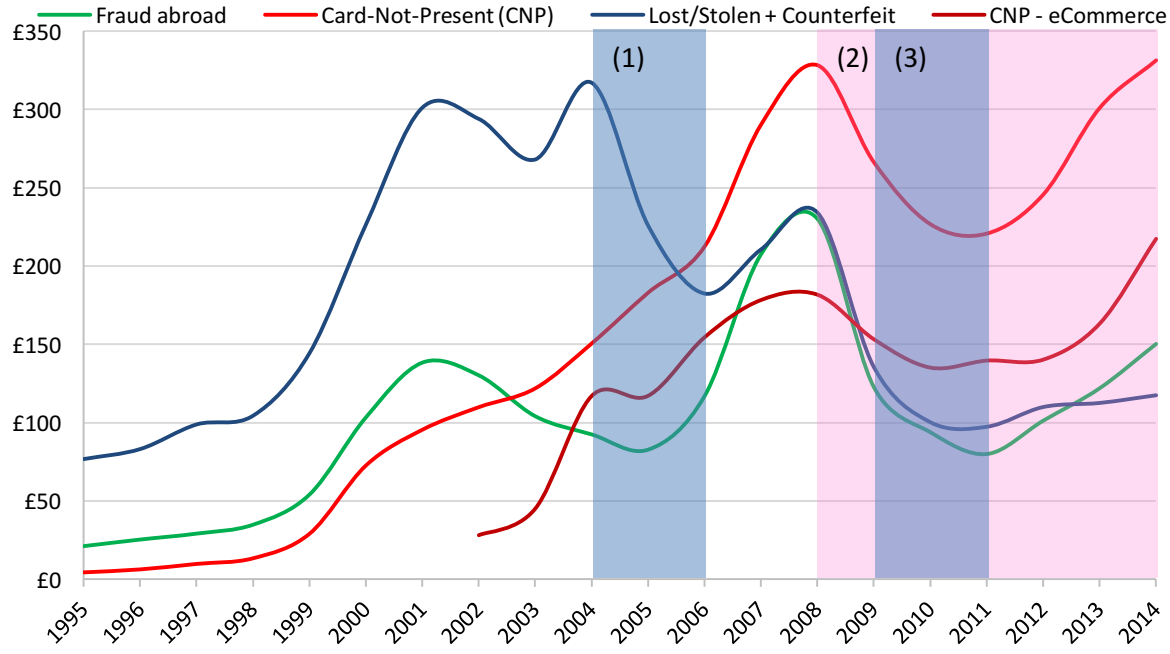


Figure 6 - UK Card Fraud by Type (£millions) [10] [11]

Figure 6 also shows the introduction of three significant security improvements to the card payments system (listed below) and their resulting impact on fraud patterns.

(1) EMV Chip & PIN cards introduced into the UK replacing the existing magnetic stripe cards. The introduction of Chip & PIN cards was phased, starting in 2004 with the majority of cards replaced by the start of 2006.

(2) 2009 reported the first year on year decrease in total card fraud reported in the UK, this decrease is mainly attributed to adoptions of sophisticated fraud screening detection tools by the Issuing banks [12]

(3) In the UK the EMV Static Data Authentication (SDA) cards were replaced by Dynamic Data Authentication (DDA) and Combined Data Authentication (CDA) cards which perform additional cryptographic authentication, making them more secure than the original SDA cards [13]

Figure 6 illustrates that each time a new security feature is introduced to the card payment system the pattern of card fraud changes. It also shows that when a new security feature is successful in reducing card fraud in one particular area, in the following years, card fraud will increase in other areas. The overall result being that the total value of card fraud has continued to steadily increase, despite a number of significant improvements in EMV card security during the period 1995 to 2014.

For example, prior to 2004 magnetic stripe payment cards were vulnerable to cloning because of the magnetic stripe, section 2.5.1, and cards authorised by signature were very vulnerable to being lost in the mail before the customer had signed the card. After 2004 the type of fraud committed moved towards cloned magnetic stripe cards being used overseas and "card not present" fraud (e.g. telephone payments). Both of these fraud types side-stepped the new Chip and PIN security features of EMV by taking advantage of weaknesses in non-EMV payment streams.

2.5.1 Magnetic Stripe Card Cloning

Magnetic stripe cards are extremely vulnerable to card cloning attacks

- the authorisation data on a magnetic stripe card is static
- magnetic stripe skimmers are small, cheap and easy to build into a malicious skimming device such as an ATM skimmer, Figure 7

Figure 7 - ATM Magnetic Stripe Skimmer [14]

Figure 7 shows an ATM card skimming device which includes both a magnetic stripe reader and a pinhole camera to capture the cardholder's PIN. The device can therefore capture all of the data required to create a cloned magnetic stripe card. Figure 7 also illustrates that the skimmer is invisible to the untrained eye when in place on the ATM.

This type of skimmer could also be used to capture the data for card-not-present fraud. Capturing the the magnetic stripe data and using the pinhole camera to capture the 3 digit CVV2 from the reverse of the card.

2.5.2 Future Trends of Card Fraud

It is hard to predict what, where and how card fraud will occur. As discussed in 2.5 the adaptable nature of fraud means whenever the banks act to implement a solution that prevents a given type of card fraud the criminals will immediately focus their attention on another weak point in the system and start exploiting that.

A good analogy would be evolution through natural selection [15], where both sides are constantly evolving, new security measures from the banks and new fraud opportunities by the fraudsters. It also follows from this analogy that there are periods when there is relatively little innovation from the banks there will be relatively little innovation from the fraudsters. The converse is also true when the banks introduce a step change in security measures such as the introduction of EMV at which point there will be a corresponding explosion in innovation by the fraudsters to find new fraud revenue streams to replace the ones lost to the new security measures; this is analogous to a major extinction event in nature when all manner of new creatures arise to fill the void after the extinction event.

Just such an extinction event is coming in the immediate future, with the phasing out of magnetic stripe cards, in those countries where it is still used. Removing magnetic stripe will create an environment in which the fraudsters are forced to innovate, as they seek out a new type of fraud which produces the maximum return for the minimum risk and effort.

2.5.3 Phase-out of Magnetic Stripe

The security features built into EMV make it difficult to use lost, stolen and counterfeit EMV payment cards in EMV compliant POS terminals and ATMs. The majority of card fraud in the UK targets the magnetic stripe on the EMV cards, which bypasses the security features provided by the EMV smartcard technology. The Magnetic stripe is present on EMV cards to support compatibility with countries where all of the ATMs and POS terminals are not EMV compliant (such as the USA).

There is a global push by the leading card scheme providers MasterCard and Visa [3] [2] [1] to replace magnetic stripe technologies with smartcard based technologies such as EMV. However, in countries where there has been a large investment in magnetic stripe technology the change to EMV represents significant investment in infrastructure, which has to date slowed the introduction of EMV into the USA, India, China and Brazil. In all of these countries the process of introducing EMV has now started with a prediction that magnetic stripe will disappear over the next 5 years (to 2020).

Once this process is complete and the magnetic stripe is removed from UK cards, as discussed in section 2.5, card fraud will adapt quickly to exploit next easiest target in the payment system. The research presented in this PhD thesis demonstrates that contactless payment cards have security weaknesses, i.e. the wireless interface and no PIN, which potentially make contactless cards the next *"low hanging fruit"* for card fraud in the UK.

2.6 Future of Payments Technologies

Payments technologies are traditionally very slow to change. This is because the introduction of any new payment technology affects the global payment network; requiring changes to the back-end payment systems at thousands of card issuing banks and changes to the payment procedures and technology at millions of shops and restaurants (merchants) accepting card payments. This makes it extremely expensive and complex to make any changes. Table 1 compares the relative lifespan of the different card payment technologies as the primary global card payment technology. It shows that (i) the rate of change is traditionally slow (ii) the speed of change is increasing (iii) there is typically a changeover period where one technology is phased out and replaced by another.

Table 1 - Card Payment Technology Life Span

Payment Technology	Operational Years	Duration
Emboss cards with imprinter and paper receipts	1950s to 1990s	40 years
Magnetic stripe credit cards	1980s to present	30 years
EMV smartcards (Chip & PIN)	2004 to present	10 years
Contactless smartcards	2009 to present	5 years
Mobile payments	2013 to present	2 years

In contrast there are currently (2015) a number of emerging payment technologies which are either being introduced / piloted or are being proposed for introduction. The model by which payment technologies are introduced is changing. Traditionally the payment network in collaboration with the card Issuing banks, have designed and introduced new payment technologies in a top-down manner, with the merchants and consumers having very little influence on the process.

New mobile phone payment technologies are being created by technology providers such as Google, Apple, PayPal and Square. The technology companies are also working directly with merchants to build the payment networks required to support the new payment systems. This fundamentally changes the way in which new technologies are introduced. Instead of a top-down introduction of a single choice technology, there is now a competition between multiple technologies which can coexist, creating a race for acceptance by consumers, the merchants and the banks. The technologies currently competing for acceptance can be categorised as shown in Table 2. Some of these technologies will gain popularity and will be adopted while others will be less popular and will disappear.

The following sections will discuss some of these technologies, look at technologies with a strong chance of being adopted and look at some interesting technology which may influence the future of payments.

Table 2 - Emerging Payment Technologies

Category	Example of the Technology	Description
Mobile Payment	Google Wallet and ApplePay	Google Wallet and ApplePay are in-store mobile payment technologies which allow you to put your existing credit / debit cards into a mobile phone based wallet.
	Zapp	Zapp is a mobile payment app which provides in-store payments and peer 2 peer payments. Several leading UK banks are integrating Zapp payment technology into their mobile banking apps
Wearable Payment Tokens	Barclays Festival Wristbands	These are wearable devices which embed the functionality of an EMV contactless payment card. The Barclays wristbands are waterproof and shock proof, perfect for festivals where carrying money and/or cards is not convenient. The wristband can make payments just like a contactless card. The band is prepaid so minimises the risk if lost and can be recharged with cash or over the Internet. The wireless interface in contactless payments allows payment technology to be integrated into just about any wearable object.
Mobile POS terminals	PayPal mobile POS and iZettle mobile POS	These are POS terminals that use a mobile phone as the connection to the payments network. The mobile POS allows mobile traders to take card payments wherever they are, e.g. the customer's premises (plumbers, builders etc.) or at trade fairs. This is a significant step forward as it enables small traders to accept card payments where previously they could only accept cash or cheques.
Peer 2 Peer Payments	PayPal Bump	This is a mobile App that allows payments to be made through the existing PayPal network. The bump technology allows two mobile phones to pair and thereby initiate the payment. This was the first app to popularise the concept of "split the bill".
	Barclay PingIT	This is a mobile online banking app with added functionality that allows peer to peer payments between two users identified by their mobile phone numbers
Mobile Banking	Barclays, HSBC, Lloyds TSB, NatWest, Santander, Nationwide, Cooperative Bank etc.	Many issuing banks provide a mobile app. However, these are essentially mobile access online banking services and not a payment app.
Biometric Authentication	MasterCard heartbeat authentication	Each person has a unique ECG pattern, which the Nymi wristband to authenticate the wearer's identity [16] [17]. Once authenticated the contactless payment can be made.
	Iris, finger-print and face recognition.	Smartphones allow payment apps to integrate various biometric authentication technologies.
Continuous Authentication	ePet movement authentication [18]	Research has been carried out into monitoring a user's gait and movements throughout the day to determine the users identity.

2.6.1 Mobile Phone Payment Devices

The vast majority of electronic payments are currently still card based. However, mobile payments are gaining popularity in countries such as the United States, Japan and South Korea where there have been large scale user acceptance and national roll outs of mobile payment solutions. Europe is lagging behind by a number of years following a number of pilot schemes which had limited user acceptance.

Even in countries like the US where mobile payments are gaining user acceptance this is still very slow. For instance card payments in the US had a total value of $4,520 billion [2] in 2013, in comparison the total value of mobile payments in the US 2013 was only $24 billion [19] which is 0.5% of the total card transactions. The worldwide total for mobile phone payments in 2013 was only $235 billion [19] which is a small total of worldwide card payments with the Gartner Group predicting that the largest growth in mobile payments will be in the area of money transfers rather than "in-store" payments [20].

The big players for mobile "in-store" payments in the United States are currently ApplePay and Google Wallet [19], with PayPal being the established player in person to person money transfers. For "in-store" payments both ApplePay and Google Wallet use NFC technology to emulate a contactless payment card. This allows the mobile payment App to be fully backward compatible with existing POS terminal technology. Unfortunately, adopting card emulation for compatibility does leave the mobile payment application open to the same vulnerabilities as the original card payments. Google Wallet and SamsungPay (LoopPay) payment Apps implement the magnetic stripe contactless transaction protocol which is the protocol used by contactless POS terminals in the United States. However, it should be noted that the EMV contactless transaction protocol [7] supports contactless magnetic stripe compatibility mode which in theory should allow European EMV POS terminals to accept payments from Google Wallet and ApplePay mobile devices.

Visa and MasterCard are pushing for merchants and issuing banks in the United States to adopt EMV smartcard technology, replacing magnetic stripe starting in 2015 [1] [3] [2]. This would mean that Google Wallet and ApplePay would should support the EMV protocol, thereby speeding up its introduction into Europe.

2.6.2 Mobile POS Terminals

Previously if a small merchant wanted to accept card payments, the only options was a POS terminal provided by a high street Bank or payments provider such as WorldPay. The merchant would be subject to credit checks, the merchant had to guarantee a minimum value of card transactions per month and there was a monthly fee for the rental of the POS terminal. All of which made it an expensive commitment to operate a POS terminal.

Traditional POS terminals require a landline telephone connection to the payments network, and for security reasons the POS terminal would be restricted to a single physical location by the bank restricting the POS to only connecting via a specific telephone number. This business model works well for shops and restaurants which have a permanent address and predictable income but is less well suited to mobile trades-people and small independent businesses.

Mobile POS terminals such as those offered by PayPal and iZettle, Figure 8, are much cheaper to purchase and operate than a high street bank issued POS terminal, with no monthly fees. This makes them a viable alternative for small traders, such as builders, plumbers and electricians, to accept credit / debit card payments at the customer's home.

The mobile POS terminal consists of a dedicated hardware module which communicates with the EMV card to processes the payment, and an App on on the merchants mobile phone which communicates with the bank to authorise the payment.

Figure 8 – Mobile POS Terminal (iZettle)

2.6.2.1 Hardware Security

The current implementations by iZettle and PayPal provide strong security because a dedicated hardware device provides a "trusted" hardware platform for all of the communications with the payment card and the cardholder. This means that the cardholder's PIN is not entered into the merchant's mobile phone, which is treated as an "untrusted platform". The mobile phone provides data communications between the POS and the Issuing Bank and provides a user interface into which the merchant enters the transaction amount.

Additional security is provided as the communications between the POS and the Issuer are encrypted meaning that the communications cannot be intercepted by the mobile phone or an internet man-in-the-middle.

2.6.2.2 Account Vetting Security

The merchant vetting process to obtain a POS terminal from the high street banks was relatively comprehensive; they required (i) that the merchant held a business bank account with the bank (ii) background checks were performed on the merchant (iii) the merchant must show a business history that justified the requirement of a POS terminal.

However, for our research we have applied for and received both an iZettle and a PayPal POS terminal, the application and vetting process was much simpler than the process enforced by the

banks. We only had to supply details for a valid personal bank account, making this type of terminal vulnerable to criminal gangs using the bank accounts of money mules.

2.6.3 Peer-2-Peer Payments

The traditional payment model is a payment requested by a merchant who is providing goods or services to a customer. Peer-2-peer payments allow payments between individuals, such as splitting a bill in a restaurant, where previously cash would have been used.

This technology is extremely convenient but does require all of the participants to have the same payments App on their mobile phone. Popular peer-2-peer payment apps include PayPal Venmo, Google wallet, Dwolla and Square Cash.

2.6.4 Mobile Banking

Mobile banking apps are provided by most of the high street banks in the UK and the US and provide functionality to allow customers to make payments such as bill pay and money transfer. However, the mechanism used to make the payments utilises the online banking service back end. The mobile app can be considered purely as a UI for the online banking service rather than an active participant in the payment activity and as such we are not considering it as part of this thesis.

2.6.5 Biometric Authentication

An accepted security axiom is that good user authentication is based on "something you are, something you know, something you have". Biometric authentication satisfies "something you are" with the PIN number being "something you know" and the embedded secure element in the mobile device or smartcard providing the "something you have".

Payment applications based on smartphone can utilise the smartphone's various built-in sensors (camera, accelerometer, microphone and touch sensors) to perform biometric authentication. Face, Iris and fingerprint recognition are popular but the accuracy depends heavily upon the quality of the hardware built into the particular model of smartphone. For example the fingerprint readers which read the ridge pattern on the finger are much easier to compromise than fingerprint readers which use the vein pattern in the fingertip to authenticate the user.

Biometric authentication is an important step forward in payments security however,

- no single biometric has proven itself to be both accurate and hard to compromise
- authentication is heavily reliant on additional hardware, not available on all mobile devices or payment tokens

2.6.6 Wearable Tokens

In the UK contactless payment technology has been incorporated into wearable payment tokens such as wristbands [21] and MasterCard have piloted contactless wristband payments technology which also incorporates continuous biometric authentication [16] [17]. These wearable payment devices will

help to promote the expansion of contactless payments, particularly in situations such as events and festivals where card payments or cash are not convenient.

2.6.7 Continuous Authentication

User authentication in payments is currently very weak being only a 4 digit PIN which is extremely easy to acquire by "shoulder surfing" the cardholder at an ATM or by coercing the cardholder into revealing their PIN. Biometric technologies can help by introducing "something you are" to the authentication process. However, biometric can suffer from accuracy issues if used in a mode where the technology is asked to authenticate right now from a single comparison.

Continuous authentication expands the use of multiple biometrics to continuously asses the identity of the user so that when the technology is asked to authenticate it has multiple data points upon which to base its decision. This greatly improves the accuracy and gives the opportunity for the authentication algorithm to compensate for individual poor quality biometric readings.

Multimodal continuous authentication uses several complementary biometric measurements to increase the accuracy of the authentication process. For instance, a mobile phone can use the accelerometers to continuously monitor the users gait and periodically (when the user turns the screen on) use the forward facing camera to perform face recognition. The results of the two biometric measures would then be combined; the influence of each periodic face recognition being high when it is captured and decaying over time until the next face recognition is performed, the accelerometer gait recognition providing a measurement that is continually topped up.

MasterCard have already run a trial with the Nymi heart monitor wristband that uses the wearers unique ECG for authentication [16] [17]. The MasterCard application uses contactless technology to communicate with the POS / ATM and make the payment. It uses multimodal authentication by requiring the user to make a specific movement to activate the payment process, at which point the wristband decides to authenticate the payment or decline based upon the most recently collected ECG data. The movement based activation process provides the secondary biometric data upon which the authentication will be based.

Multimodal Continuous authentication techniques also have the option of asking the user to provide PIN authentication for the payment covering "something you know". In the case of a payment wristband the wristband itself becomes the "something you have", to do this the wristband must have a secure element / uniquely identifiable chip, embedded into it.

2.7 Conclusion

Contactless payments are the first of many new payment technologies being introduced to the EMV payment system. By analysing the vulnerabilities that contactless payments have introduced we can better understand how those vulnerabilities impact the patterns of card payment fraud in the future. The new technologies, described in this section, will both combat fraud and create new opportunities for fraud, as shown by Figure 6.

In the following sections I analyse the EMV transaction protocol and identify vulnerabilities introduced to the protocol by the new contactless payment technology. I outline an analysis methodology that can be used to analyse future changes to the EMV protocol required for the introduction of subsequent payment technologies and identify potential vulnerabilities.

Chapter 3. EMV Transaction Protocol

This chapter describes the operation of the EMV transaction protocol. The protocol is a sequence of messages exchanged between a POS terminal / ATM and an EMV card, requesting payment authorisation from the EMV payment card. The transaction protocol sequence consists of a number of operations; selecting the card, requesting the payment, checking that the card is valid, checking that the cardholder is authorised to use the card and authorising the payment.

The EMV specifications [6] [7] comprise 2392 pages and describe the data and operations required to perform the transaction steps. However, the specifications do not contain a single definitive description of the EMV transaction protocol sequence. Rather they contain a number of optional protocol sequence elements which can be tailored to the banking regulations in different countries and proprietary requirements of the different card types (MasterCard, Visa, American Express, JCB, Discover and UnionPay).

3.1 Variants of the EMV Transaction Protocol

There are nine variants of the EMV transaction protocol sequence in the EMV specifications dependent upon the card type and the payment method (Chip & PIN or contactless Figure **2**):

- contact "Chip & PIN" transaction protocol
- kernel 1 - Visa contactless transaction protocol with online transaction authorisation support
- kernel 2 - MasterCard contactless transaction protocol
- kernel 3 – Visa fDDA contactless transaction protocol for offline only transactions
- kernel 4 - American Express contactless transaction protocol,
- kernel 5 – JCB contactless transaction protocol
- kernel 6 - Discover contactless transaction protocol
- kernel 7 – UnionPay contactless transaction protocol.
- contactless magnetic stripe transaction protocol

These variants can be markedly different, for instance kernel 3 (Visa) fDDA is designed to always complete the transaction without requiring authorisation from the issuing bank whereas the kernel 2 (MasterCard) always requests authorisation from the card issuing bank but may complete if the Issuing bank is unavailable. The differences between the two transaction protocols has a direct impact on the security of Visa contactless cards. The high value contactless foreign currency flaw,

described in Chapter 6, is only applicable to the kernel 3 fDDA protocol. Kernels 1,2,4,5 and 6 would detect the attack when the protocol requests online transaction authorisation from the Issuing bank.

3.2 Structure of EMV Transaction Protocol

All transaction protocol sequences comprise three stages;

- card authentication
- cardholder verification
- transaction authorisation

Each of the three stages has a number of options dependent on the capabilities of the of the card and of the POS terminal / ATM.

Card Authentication: in the EMV transaction the POS terminal / ATM authenticates the card by requesting a digital signature from the card. EMV supports three methods of generating the signature Static Data Authentication (SDA), Dynamic Data Authentication (DDA) and Combined Data Authentication (CDA).

Cardholder Verification: during the transaction the POS terminal / ATM confirms the identity of the cardholder. The method used to verify the cardholder is dependent on the type of card and the type of transaction being performed. The options are, the cardholder enters their PIN, the cardholder provides a signature on paper or, in the case of contactless transactions, there is no cardholder verification.

Transaction Authorisation: on completion of the transaction the card generates an Application Cryptogram (AC) which encodes the final authorisation decision, accept or decline, along with the details of the transaction. There are two options for transaction authorisation, online and offline. Offline authorisation is performed by the card alone. During online authorisation the POS / ATM connects to the Issuing bank, authorises the transaction by producing an Issuer Application Cryptogram, which the card then countersigns to confirm Issuers transaction authorisation.

The options in transaction protocol allow the EMV protocol to support interoperability between the smartcards, POS terminals and ATMs across the 76 countries currently operating EMV payments. The options also allow the card scheme providers to include proprietary functionality specific to their cards within the transaction protocol.

Figure 9 shows the three stages of the transaction protocol sequence and the options available within each stage, the diagram illustrates the complexity of the transaction protocol sequence. One of the assertions of this PhD thesis is that this complexity can lead to reduced security in the EMV transaction protocol.

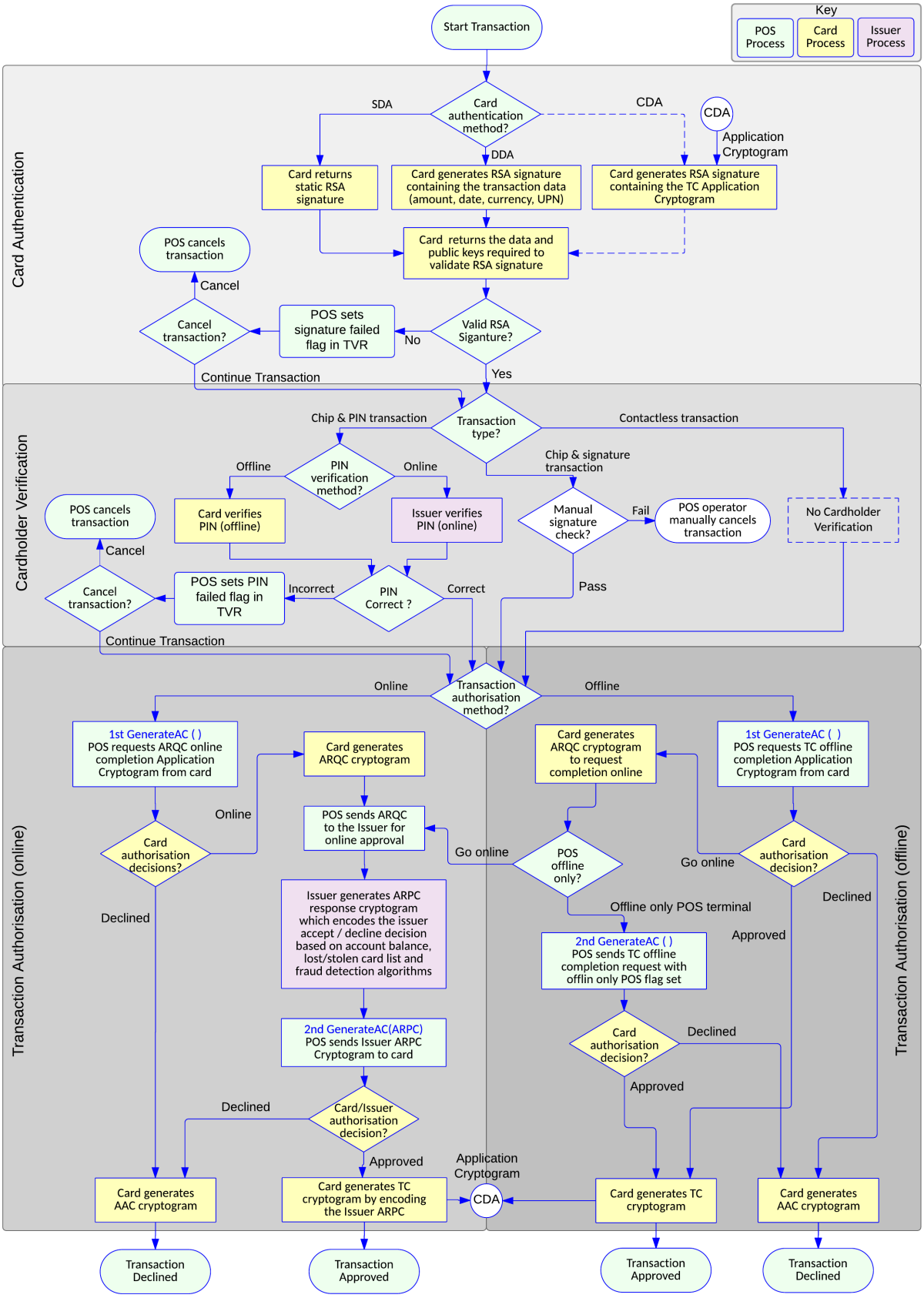


Figure 9 - EMV Transaction Protocol

3.3 Card Authentication

EMV supports three distinct card authentication methods, Static Data Authentication (SDA), Dynamic Data Authentication (DDA) and Combined Data Authentication (CDA) [6] [22]. All of which use an RSA digital signature to authenticate the card. The POS terminal / ATM authenticates the digital signature provided by the card using its own copy of the Certificate Authority (CA) public key applicable for that card type, see Figure 10 and Figure 11.

The choice between the different authentication methods, SDA, DDA and CDA, has, to a certain extent, been driven by the processing power available on the card to perform the cryptographic authentication process necessary for the different authentication methods:

SDA the cards return a static RSA digital signature, generated by the Issuer and written to the card during manufacture, which requires no cryptographic processing by the card. SDA cards only support plaintext PIN verification, which does not require any cryptographic processing.

DDA the card dynamically generates an RSA digital signature of the transaction data. DDA cards support both plaintext and enciphered PIN verification. The PIN is RSA enciphered by the POS terminal / ATM using the card's own public key.

CDA the card dynamically generates an RSA signature of the Application Cryptogram (AC). CDA cards support both plaintext and enciphered PIN verification.

All three card types (SDA, DDA and CDA) support the cryptographic processing required to generate and validate Application Cryptograms in the Transaction Authorisation phase. The Application Cryptograms are an 8-byte Message Authentication Code (MAC) generated over the transaction data using either the 3-DES or AES algorithm.

In the UK the earliest EMV cards, issued in 2004, were SDA cards which required the least processing power. By 2009 the processing power of smartcards had increased and the cost had decreased which allowed SDA cards to be replaced by DDA cards and CDA cards. In the UK, since 2009, the majority of cards are DDA, however, in countries such as Poland SDA cards are still very common.

3.3.1 Static Data Authentication (SDA)

The POS terminal authenticates an SDA card by verifying the RSA digital signature provided by the card. Figure 10 shows the SDA card authentication process. The SDA card has a static signature which is generated by the Issuer, signed using the Issuers private key, and written to the SDA card during manufacture. The Issuer public key / private key pair are generated by the CA using the CA's private key. The SDA digital signature is validated using the CA public key stored on the POS terminal / ATM and the Issuer public key provided by the card.

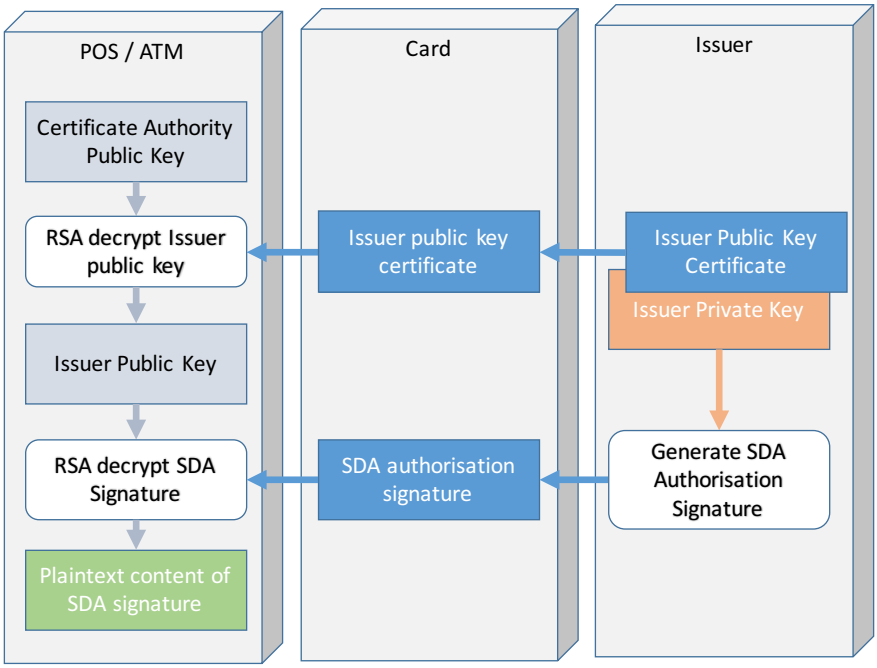


Figure 10 – SDA card authentication process

See section 3.3.4 for further details of the RSA Public Key Infrastructure (PKI) used for SDA, DDA and CDA cards.

Vulnerabilities of Static Data Authentication (SDA)

SDA payment cards are the simplest and the cheapest card variant supported by EMV, it is also considered the least secure. The SDA cards are considered to be an improvement on the magnetic stripe cards they replaced because of the physical security features explained in section 2.3. However, SDA cards, have a static authorisation signature which is used to approve every transaction, this makes SDA cards vulnerable to cloning in attacks such as the YES card attack [23].

3.3.2 Dynamic Data Authentication (DDA)

DDA cards generate a unique RSA signature (SDAD) for each transaction. The digital signature encodes the transaction data and an unpredictable number generated by the POS terminal / ATM for each transaction, Table 3. This prevents an attacker from recording the SDAD signature and creating a cloned card because the SDAD changes for each new transaction. Encoding the transaction data in the digital signature also serves to ensure that transaction data has not been changed from that requested by the POS terminal / ATM, thereby preventing man-in-the-middle attacks changing transaction data, such as amount.

Table 3 details minimum recommended data which should be included in the DDA digital signature. Issuers may include additional fields. However, in our experiments (Chapter 5, Chapter 6, Chapter 7 and Chapter 9) the majority of cards tested were kernel 3 DDA cards, which produced SDAD signatures with only the fields detailed in Table 3.

Table 3 – SDAD minimum recommended data fields

Data Field	Purpose of the Data
Transaction Amount	Ensures that the card is validating the correct value of transaction
Transaction Date	Ensures that the card is validating the correct date of transaction
Unpredictable Number (POS / ATM)	Random data to prevent cloned cards using pre-recorded digital signatures in pre-play attacks
Unpredictable Number (card)	Random data inserted by the card to prevent plaintext oracle attacks on the card's private key.

Figure 11 illustrates the process of validating the DDA signature. The POS terminal / ATM uses its own copy of the CA public key to validate the Issuer public key certificate provided by the card. The resulting Issuer public key is used to validate the card's public certificate. The card's public key is used to validate the DDA authentication signature (SDAD).

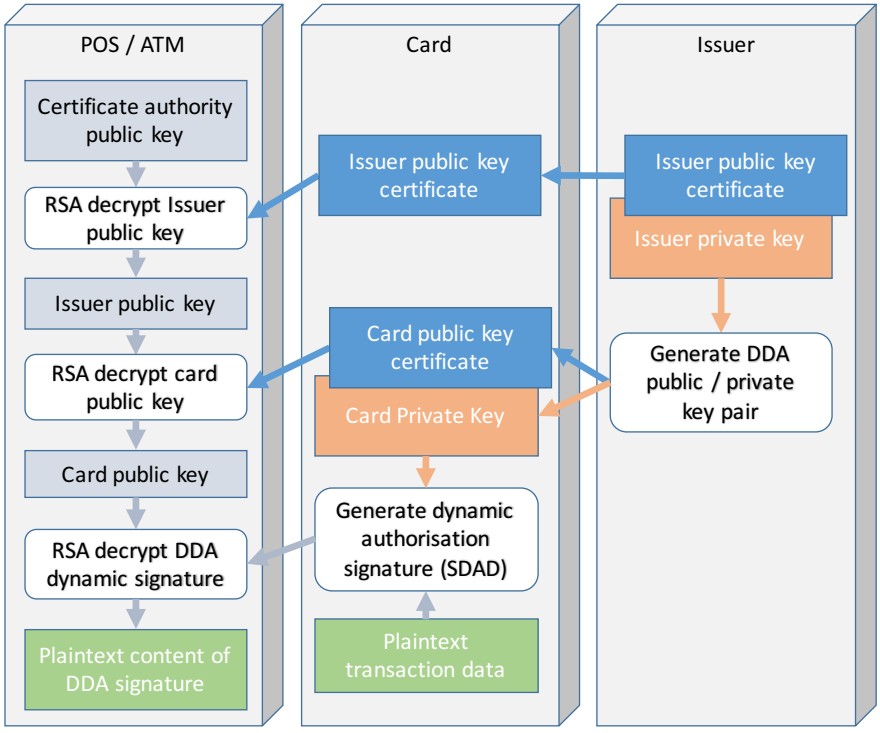


Figure 11 – DDA card authentication process

See section 3.3.4 for further details of the RSA Public Key Infrastructure (PKI) used for SDA, DDA and CDA cards.

Vulnerabilities of Dynamic Data Authentication (DDA)

The data included in the DDA dynamic signature, Table 3, changes for each transaction. This protects DDA cards from cloning attacks and pre-play attacks. There are theoretical attacks on DDA signature generation, Degabriele et al. (2012) [24] describes and attack on the RSA cryptography and Bond et

al. (2014) [25] describes an attack on and the random number generation. These attacks are possible but are impractical and therefore unlikely to be exploited, as discussed in Chapter 4.

DDA cards are vulnerable to a number of attacks which down grade the transaction authorisation options described in Figure 9. For instance, Murdoch et al. (2010) [13] identifies a security weakness which exploits the options available in the cardholder verification stage. The POS terminal is tricked into believing that the card has verified the cardholder's PIN and the card believes that the cardholder has been verified by signature on paper. Having multiple options of cardholder verification options allows a man-in-the-middle to skip the cardholder verification stage by manipulating both sides of the transaction (POS and card) into believing the step has already been performed.

3.3.3 Combined Data Authentication (CDA)

CDA authentication is a variation of DDA authentication. Just like DDA, the CDA cards generate a dynamic signature which is different for each transaction. CDA improves upon DDA by encoding the Application Cryptogram (AC) into the signature rather than the transaction data. This links the SDAD signature used by the POS terminal / ATM to verify the transaction, with the AC used by the Issuer to verify the transaction.

The process of authenticating a CDA digital signature is exactly the same as the authentication process for DDA cards, Figure 11, only the data encoded in the signature is changed. The data included in the CDA signature includes the AC, it may also include the transaction data fields included in the DDA digital signature Table 3, thereby providing an additional security check.

Vulnerabilities of Combined Data Authentication (CDA)

CDA is designed to prevent a specific category of offline "wedge" attacks which exploit the fact that in a DDA transaction, the issuer validates the AC while the POS terminal validates the SDAD, neither validates both the CA and the SDAD together. However, CDA would still be vulnerable to down grade attacks such as Murdoch et al. (2010) [13] and Roland and Langer (2013) [26].

3.3.4 RSA Public Key Infrastructure (PKI) for SDA, DDA and CDA cards

Validating the RSA digital signatures, generated by SDA, DDA and CDA cards, relies upon a three-tier PKI; CA => Issuer => card. The CA is the card scheme provider, Visa, MasterCard, American Express, JCB, Discover or UnionPay, of the card. The CA signs the Issuer's public key / private key pair, which then allows the Issuer to generate SDA digital signatures and RSA public / private key pairs for DDA / CDA cards. The card's private key is used to validate the RSA digital signature.

SDA card authentication, Figure 10, and DDA / CDA card authentication, Figure 11, both start with the POS terminal / ATM authenticating the Issuer public key provided by the card. The POS / ATM uses it own copy of the CA public key to ensure that the key provided by the card is issued by a recognised issuing bank.

In SDA card authentication the Issuer public key is then used to validate the SDA digital signature.

In DDA / CDA the issuer public key is used to validate the card's public key, which is then used to validate the DDA / CDA digital signature.

The issuer public key and card public keys have expiry dates encoded into them which limit the lifespan of the keys. The card public key includes the 16 digit card number (PAN) and the Issuer public key contains the Issuer Identifier, which allows the POS terminal to compare these values to the plaintext equivalent provided by the card. This ensures that the card data matches the digital signature used to authenticate the card.

3.4 Cardholder Verification

Cardholder verification is governed by the capabilities of the card, the capabilities of the POS terminal and upon the type of transaction being performed. The card publishes a list of its capabilities in the Cardholder Verification Method (CVM). The CVM list also specifies the order of priority in which the POS terminal should select the cardholder verification method to be used.

The options for cardholder verification are

- PIN entry by the cardholder
- cardholder signature on paper
- no cardholder verification (used for contactless payments).

3.4.1 Cardholder verification by PIN

This is the preferred method of verifying the cardholder in contact "Chip & PIN" transactions. The cardholder enters their PIN into the PIN pad on the POS terminal / ATM. The PIN can be validated by the Issuer (online) or by the card (online). Note; this is independent of the online / offline transaction authorisation described in section 3.5.

Online PIN verification performed by the issuer; the POS terminal / ATM transmits the PIN entered by the cardholder to the Issuer for verification. The PIN is RSA enciphered prior to transmission to the Issuer using the Issuer public key provided by the card. Online PIN verification is the preferred method for SDA cards as this prevents the YES card attack.

Offline PIN verification performed by the card; the PIN is sent to the card for verification. The PIN can be transmitted in plaintext or RSA enciphered using the card's public key. The card responds with PIN OK or, if the PIN was incorrect, an error code which includes the number of PIN attempts remaining.

Plaintext PIN; the EMV specification does not allow online PIN verification, as this would mean that the PIN was transmitted over public networks unencrypted. SDA cards only support plaintext PIN verification as the card does not support RSA encrypt, decrypt, sign and verify functionality.

Enciphered PIN; the PIN is RSA enciphered for both online verification by the issuer and offline verification by the card. SDA cards do not support enciphered PIN. DDA and CDA both support enciphered PIN.

Contactless cards do not require PIN verification functionality as contactless transactions do not require cardholder verification. Contactless transactions should always use the no cardholder verification option in Figure 9. However, our research in Chapter 7 shows that many Visa cards support contactless PIN verification. This demonstrates that the multiple options in protocol can be exploited to produce undesired functionality [27].

3.4.2 Cardholder verification by signature

Cardholder verification by PIN is an option available on SDA, DDA and CDA cards. The CVM list on the card determines if the card supports verification by signature. The CVM list is structured in such a way that if the card and terminal support PIN verification this will always take priority over verification by signature.

Verification by signature is only available at attended POS terminals. The process involves the POS terminal printing the merchant receipt, the cardholder signs the receipt, the attendant checks the signature and presses a button on the POS terminal to indicate that the signature is approved.

3.4.3 No cardholder verification

Contactless transactions do not require the cardholder to be verified by PIN or by signature. The trade-off being that contactless transactions are restricted to low value transactions.

"No Cardholder Verification" is intended for contactless transactions only, therefore, contact "Chip & PIN" transactions should not be able to select this option in Figure 9. No cardholder verification is always the lowest priority option in the CVM list and therefore the last to be selected if no other CVM options are available.

3.5 Transaction Authorisation

Transaction authorisation is achieved in a bidirectional exchange of Application Cryptogram between the card and the Issuer, with the POS terminal / ATM acting as the communication channel between the card and the Issuer. The Transaction Authorisation stage in Figure 9 shows the sequence and flow of cryptograms between the Issuer and the card.

The cryptograms encode the outcome of the transaction authorisation decision. There are a number of different types of cryptogram which can be generated by the card or the Issuer:

TC *Transaction Certificate*, is generated by the card, indicating that the transaction is approved. The TC is a cryptographically secure token which the POS / ATM sends to the Issuer as proof that the transaction was approved by a genuine card. The TC is sent along with the

transaction details allowing the bank to validate the transaction data supplied by the POS / ATM against the transaction approved by the card.

ARQC *Authorisation Request Cryptogram*, is generated by the card to request online confirmation of the transaction by the Issuer.

APRC *Authorisation Request Cryptogram*, is generated by the Issuer in response to ARQC produced by the card and indicates the Issuer's authorisation decision to approve or decline the transaction.

AAC *Application Authentication Cryptogram*, is generated by the card to indicate that the transaction was declined.

The POS / ATM sends a Generate AC command to the card to request that the card generates a cryptogram. The command includes the type of cryptogram being requested TC, ARPC or AAC to indicate the desired response from the card.

The card's response is dependent on the type of cryptogram being requested and the order of precedence (TC => ARQC => AAC). The card will only return a cryptogram of the same or lower precedence than that requested by the POS / ATM. The card will respond as detailed in Table 4

Table 4 – First Generate AC - Application Cryptogram Request / Response Mapping

Cryptogram requested by POS / ATM		**Response of the card**	
TC	the POS terminal / ATM request final transaction authorisation	TC	transaction approved, no further action.
		ARQC	the card request that requests online completion of the transaction. The ARQC also encodes the transaction data to be send to the Issuer.
		AAC	transaction is declined, no further action.
ARQC	the POS terminal / ATM requests online completion of the transaction.	ARQC	the card generates an ARQC cryptogram encoding the transaction data to be send to the Issuer.
		AAC	transaction is declined, no further action.
AAC	the POS terminal / ATM requests a transaction declined cryptogram.	AAC	the card generates AAC transaction declined to send to the Issuer.

The transaction authorisation stage Figure 9 includes two Generate AC commands. In the second Generate AC command the POS terminal / ATM sends the ARPC generated by the Issuer to the card. The ARPC encodes the Issuers transaction approve / decline decision. The card decrypts the ARPC and responds with the appropriate cryptogram, TC if the Issuer approved the transaction and ACC if

declined. The second Generate AC command therefore has a limited set of possible requests and possible responses as detailed in Table 5.

Table 5 – Second Generate AC – Application Cryptogram Request / Response Mapping

Cryptogram requested by POS / ATM		Response of the card	
ARPC	the POS terminal / ATM send the ARPC generated by the Issuer.	TC AAC	transaction approved. transaction is declined.
AAC	the POS terminal / ATM requests a transaction declined cryptogram.	AAC	the card generates AAC transaction declined to send to the Issuer.

3.5.1 Transaction Authorisation Modes

Transaction Authorisation has two modes. In offline mode the card alone authorises the payment. In online mode the card issues an ARQC cryptogram to request authorisation from the Issuer, the issuer responds with the ARPC authorisation decision cryptogram which the card confirms returning the final transaction authorisation cryptogram TC or AAC to the POS terminal / ATM.

Figure 9 illustrates that both the POS terminal / ATM and the card are involved in the decision to complete the transaction in online mode or offline mode. The POS terminal / ATM can request online completion in which case the transaction will be completed online. If the POS terminal / ATM requests offline completion the card can override the request by requesting online completion, in which case there is a negotiation between the card and the POS terminal / ATM.

Offline mode transaction authorisation

In offline transaction authorisation the POS / ATM requests a final authorisation decision, TC cryptogram, from the card on the first Generate AC request. If the card is willing to authorise the transaction offline, it generates the TC cryptogram and the transaction is complete. In this, fully offline case, the Issuer is not involved in authorising the transaction.

Online mode transaction authorisation

Online transaction completion can be requested by either the POS terminal / ATM or by the card. The POS terminal / ATM requests online completion of the transaction by sending ARQC in the first Generate AC command. If the POS terminal / ATM requests offline completion of the transaction, the card can request online completion by replying to the POS terminal / ATMs first Generate AC command with an ARQC cryptogram.

Whether requested by the POS / ATM or by the card the ARQC produced by the card is sent to the Issuer. The issuer verifies the ARQC produced by the card and responds with an ARPC containing the Issuers final accept / decline decision. The POS terminal / ATM passes the Issuer ARPC to the card in the second Generate AC command. The card will then produce either a TC if the Issuer accepted the transaction or an AAC if the Issuer declined.

3.5.2 Application Cryptogram Data and Encryption

The cryptograms (ARQC, ARPC, TC and AAC) which pass between the card and the Issuing Bank are 3-DES encoded using a shared symmetric key [28]. The POS / ATM simply passes the messages from the card to the Issuer and back, the POS / ATM cannot read or even verify the authenticity of the cryptograms as this requires the symmetric key which only the card and the Issuer possess.

The 3-DES encoded cryptograms are 8 bytes in length and contain the transaction data, Table 6, and the type of the cryptogram either ARQC, ARPC, TC or AAC. Encoding these fields into the cryptogram ensure that the details of the transaction cannot be changed from those approved by the Issuer / card. As with the SDAD the AC contains unpredictable / random data to prevent card cloning and pre-play attacks.

Table 6 - Application Cryptogram Minimum Recommended Data Fields

Value	Source	Comment
Transaction Amount	POS / ATM	The 3-DES encoding of the AC by the card ensures that the transaction amount cannot be altered by a man-in-the-middle from that approved by the card.
Amount Other	POS / ATM	Amount other is typically used for cashback at POS terminals (e.g. groceries "transaction amount" = £43.27 and cashback "amount other" = £20.00)
Terminal Country	POS / ATM	Location of the POS / ATM.
Terminal Verification Results (TVR)	POS / ATM	The TVR contains a record of the authorisation processes performed by the POS / ATM. However, the TVR only records failure conditions it does not record the steps which were successfully carried out.
Transaction Currency	POS / ATM	Currency can be different from country
Transaction Date	POS / ATM	
Transaction Type	POS / ATM	
Unpredictable Number	POS / ATM	The transaction data includes a 4 byte random number which prevents replay attacks
Application Interchange Profile (AIP)	ICC (card)	
Application Transaction Counter (ATC)	ICC (card)	The inclusion of the card's transaction counter prevents the cryptogram from being used to authorise multiple payments.

3.6 Contactless Transaction Protocol

The EMV contactless payment specifications [7] include seven variants of the contactless transaction protocol; kernel 1 - Visa, kernel 2 - MasterCard, kernel 3 - Visa fDDA, kernel 4 - American Express, kernel 5 JCB, kernel 6 Discover and kernel 7 – UnionPay fDDA).

Within these seven variants there are two distinctly different protocol sequences;

- kernels 3 and 7 use the fast Dynamic Data Authentication (fDDA) protocol which is an offline only transaction authorisation protocol. fDDA is designed to minimise the time required to process the transaction and thereby minimise the time the card must be present in the POS terminal[1] NFC field.
- kernels 1, 2, 4, 5 and 6 contactless transaction protocols more closely follow the generic EMV transaction protocol sequence, Figure 9, used in "Chip & PIN" transactions. They support online and offline transaction authorisation and CDA and DDA card authentication.

For brevity I will refer to two kernels which are typical examples of the different protocol sequences; kernel 3 (Visa) to represent the fDDA protocol and kernel 2 (MasterCard) to represent online / offline capable transaction protocols.

Both protocol sequences, kernel 3 and kernel 2, follow the structure of the transaction authorisation sequence described in Figure 9, having a card authentication stage, a cardholder verification stage and a transaction authorisation stage.

Card Authentication

Kernel 3 supports DDA card authentication only, whereas, kernel 2 supports both DDA card authentication and CDA card authentication.

Cardholder Verification

Contactless payments are designed for speed and convenience, therefore do not require the cardholder to spend time entering a PIN or signing for the transaction. Both kernel 2 and kernel 3 support this using the "no cardholder verification" option in Figure 9.

Transaction Authorisation

This is where the differences between kernel 2 and kernel 3 are most significant. The transaction authorisation process used in kernel 2 is the same at that described in Figure 9. Kernel 2 has full support for both online and offline transaction authorisation. The kernel 2 protocol includes two

[1] Contactless payments valid for purchases at POS terminals only, ATM cash withdrawals are not currently part of the EMV contactless specification.

Generate AC commands which allows the full exchange of Application Cryptograms between the card and the Issuer.

Kernel 3 fDDA only supports offline transaction authorisation. The card will either approve the transaction offline or decline the transaction based on the value of this transaction and the cumulative value of the offline transactions performed by the card since the last online transaction.

Kernel 3 simplifies the protocol down to a single command, Get Processing Options, which passes the transaction details to that card. Kernel 3 combines the card authentication stage, producing the SDAD, and the transaction authorisation stage, generating the final TC (transaction approved) cryptogram into this single command.

If a kernel 3 card responds by generating ARQC (online request) or AAC (transaction declined) the kernel 3 fDDA transaction is ended and the POS terminal must run a "Chip & PIN" contact transaction. There is no option in kernel 3 fDDA for online completion of the transaction.

3.6.1 Comparison between "Chip & PIN", kernel 3 and kernel 2 protocols

In this section the diagrams compare the Chip & PIN transaction protocol, Figure 12, kernel 2 transaction protocol, Figure 13, and kernel 3 transaction protocol, Figure 14.

The Chip & PIN transaction protocol implements all three stages of the generic EMV transaction protocol, card authentication, cardholder verification and transaction authorisation. Figure 12 shows a Chip & PIN transaction with the online transaction authorisation .

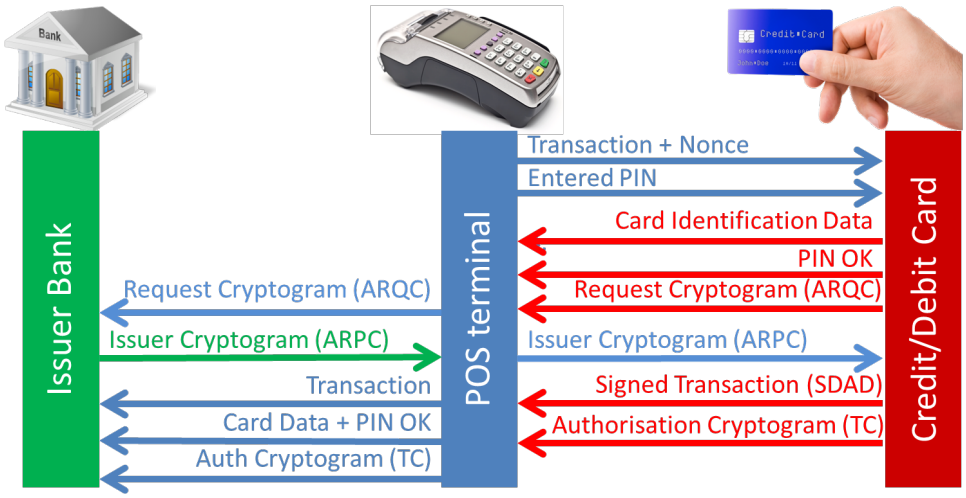


Figure 12 - Chip & PIN Transaction Protocol (online)

The kernel 2 contactless protocol sequence, Figure 13, closely follows the Chip & PIN transaction protocol. The only significant difference between being that PIN verification process is not required in the contactless transaction protocol. In Figure 13 the messages involved in PIN verification are greyed out to indicate that they are not present in the kernel 2 transaction protocol.

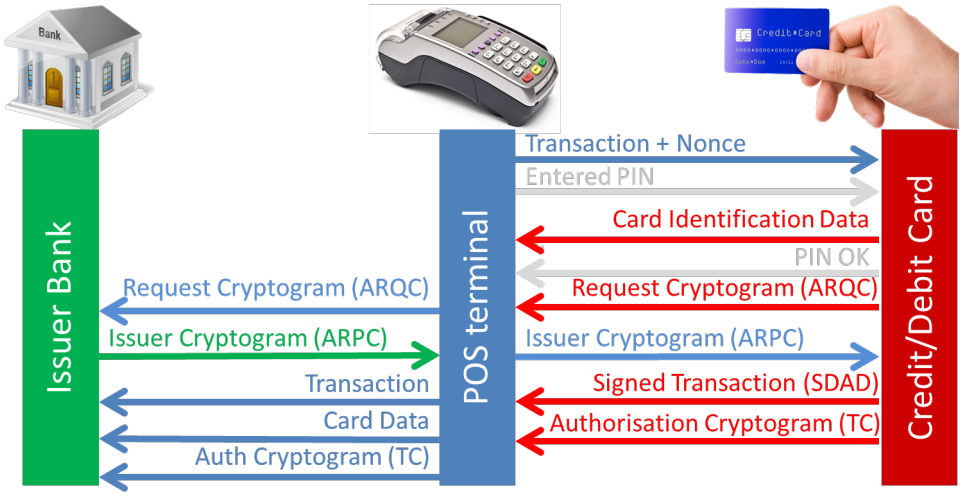


Figure 13 - Kernel 2 Contactless Transaction (online)

The kernel 3 fDDA transaction protocol, Figure 14, is fundamentally different from kernel 2 and the generic EMV "Chip & PIN" transaction protocol. The protocol is stripped down to a single command, Get Processing Options, which sends the transaction data plus a random nonce to the card. The card responds with the card authentication SDAD, the TC (transaction approved) cryptogram and the plaintext card data required by the Issuer. This completes the transaction.

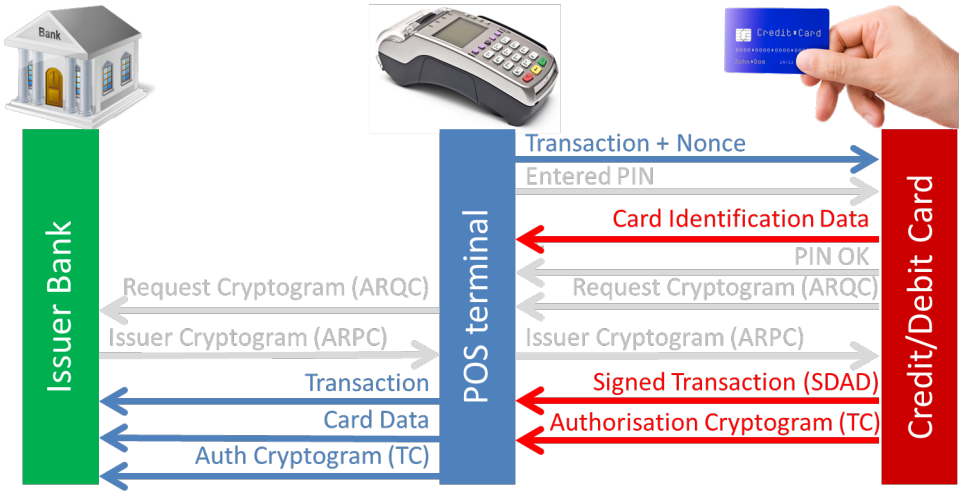


Figure 14 – Kernel 3 fDDA Contactless Transaction (offline only)

The kernel 3 protocol sequence is very fast, completing in less than 500ms in our experiments [29]. However, to achieve the fast contactless transaction times the protocol sequence drops the authorisation request cryptogram (ARQC) and response from the issuer (ARPC). This reduces the capability of the Issuer to catch fraudulent transactions.

One of the identified impacts of this difference between the two protocols is that kernel 3 (Visa) contactless cards are vulnerable to the foreign currency flaw, discussed in Chapter 6, whereas kernel 2 (MasterCard) contactless cards are not vulnerable.

3.7 Magnetic Stripe Contactless Transaction Protocol

The magnetic stripe contactless transaction protocol is an alternative to the EMV contactless protocol, implemented in countries such as the USA who have implemented contactless payments in a magnetic stripe payment environment.

A magnetic stripe contactless transaction simulates swiping a magnetic stripe card through the magnetic stripe reader in the POS terminal.

The POS terminal sends the Get Processing Options command to the contactless card. The card responds with all of the data required to replicate a magnetic stripe; the track 1 data, the track 2 data and the dynamic CVV.

Kernels 2,3,4 and 6 include support for magnetic stripe contactless protocol for compatibility with countries still using magnetic stripe. This makes EMV contactless cards vulnerable to the combined pre-play / downgrade attack described by Roland and Langer (2013) [26].

3.8 Conclusion

This brings us back to our original question *"has usability been improved at the cost of security?"*. EMV is extremely flexible because, as Figure 9 illustrates, EMV security is designed as a set of optional authentication processes. However, it is these options that make the EMV transaction protocol vulnerable to attacks which *"downgrade"* the transaction protocol to a lower security option, such as, Murdoch et al. (2010) [13] and Roland and Langer (2013) [26] and Barisani et al. (2011) [30].

Chapter 4. Literature Review of EMV Protocol Security Vulnerabilities

This literature review presents research related to analysis of the security of the EMV protocol and the underlying smartcard technologies. The review is divided into four categories:

- structured / formal analysis of the EMV protocol
- exploitable vulnerabilities in the EMV protocol
- exploitable vulnerabilities in EMV contactless technology
- review of known vulnerabilities in the EMV payment system.

In this literature review I draw out the recurring issues, identified in the introduction, the wireless interface, PIN entry not required and the increased complexity due to the seven different contactless kernels. This will be done by relating these issues to the selected research papers discussed in this chapter:

4.1 Research Categories

This section lists the academic research papers included in this literature review and identifies which of the four categories to which they are applicable.

Structured / Formal Analysis of the EMV Protocol

De Ruiter and Poll (2011) "Formal Analysis of the EMV Protocol Suite"

De Koning Gans and De Ruiter (2012) "SmartLogic tool: Analysing & testing smartcard protocols"

Choudary (2010) "The Smart Card Detective: A HandHeld EMV Interceptor,"

Aarts (2013) "Formal models of bank cards for free"

Ouerdi et al. (2013) "Abstract tests based on SysML models for EMV Card".

Pasquet et al. (2008) "Secure Payment with NFC Mobile Phone in the SmartTouch Project"

Ming-hui et al. (2007) "Security Mechanism Research of EMV2000

Exploitable Vulnerabilities in the EMV Protocol

Murdoch et al. (2010) "Chip & PIN is Broken"

Roland and Langer (2013) "Cloning credit cards: a combined pre-play and downgrade attack on EMV contactless"

Bond et al. (2014) "Chip and Skim: cloning EMV cards with the pre-play attack"

Barisani et al. (2011) "Chip & PIN is definitely broken"

Degabriele et al. (2011) “On the Joint Security of Encryption and Signature in EMV”

Anderson et al. (2005) “Chip & SPIN”

Emms et al. (2014) “Harvesting High Value Foreign Currency Transactions from EMV Contactless Credit Cards Without the PIN”

Emms et al. (2013) “Risks of Offline Verify PIN on Contactless Cards.”

Exploitable Vulnerabilities in EMV Contactless Payment Technology

Francis et al. (2012) “Practical Relay Attack on Contactless Transactions by Using NFC Mobile Phones”

Roland and Scharinger (2013) “Applying Relay Attacks to Google Wallet”

Roland et al. (2012) “Relay Attacks on Secure Element-Enabled Mobile Devices: Virtual Pickpocketing Revisited”

Roland et al. (2012) Practical Attack Scenarios on Secure Element-Enabled Mobile Devices

Kfir and Wool (2005) “Picking Virtual Pockets using Relay Attacks on Contactless Smartcard Systems”

Diakos et al. (2015) “Eavesdropping near-field contactless payments: a quantitative analysis”

Hancke (2011) “Practical Eavesdropping and Skimming Attacks on HF RFID Tokens.”

Kirshenbaum and Wool (2006) “How to Build a Low-Cost Extended-Range RFID Skimmer”

Oren et al. (2013) Range Extension Attacks on Contactless Smart Cards

Review of Known Vulnerabilities in the EMV Payment System

Markantonakis et al. (2009) Attacking smart card systems: Theory and Practice

Anderson (2007) “RFID and the middleman”

Anderson et al. (2005) “Chip & SPIN”

Henzl (2012) “Security of Contactless Smart Cards”

Anderson (1993) “Why Cryptosytems Fail”.

Anderson et al. (2012) “Might Financial Cryptography Kill Financial innovation?”

Murdoch and Anderson (2014) “Security Protocols and Evidence: Where Many Payment Systems Fail”.

Anderson and Moore (2013) “The Economics of Information Security and Privacy”

Moore and Anderson (2011) “Economics and Internet Security”

4.2 Structured / Formal Analysis of the EMV Protocol

The primary contributions of this PhD thesis are the analysis methodology of the EMV protocol, presented in Chapter 5 and the previously undocumented vulnerabilities in the EMV protocol, which were identified using the analysis methodology and are presented in Chapter 6 and Chapter 7. There is a limited amount of published research into structured analysis of the EMV protocol itself, our analysis is therefore a significant new addition to this research area.

De Ruiter and Poll (2011) [28, 31] is the most comprehensive formal analysis of the EMV protocol. They have built an F# abstract model of the EMV contact (Chip & PIN) protocol and use the ProVerif verification tool to perform analysis. This research provides a very detailed model of the EMV protocol. For instance, both their F# model contains detailed descriptions for low level data objects such as the CVM list, the PDOL, the AFL and AIP. These low level data objects become the pre-conditions of the transaction in the F# model, with the model attempting to resolve all of the possible starting values of these data objects to find out if any of the combinations of starting values end in an unexpected outcome and thereby point to a vulnerability in the EMV protocol. Their F# model also depicts the permutations the different modes of the EMV transaction are fully modelled i.e. online / offline, PIN verified / signature verified and SDA / DDA / CDA authentication.

The future work in this paper states that the F# model could be compiled to run as an application. This would allow the F# model to act like our protocol emulator code and interact with real EMV cards, thereby combining a formal model and practical experiments into one code base.

The results of this analysis show that the model detects existing known issues in the Chip & PIN protocol such as Murdoch et al. (2010) [13]. However, it did not highlight the known error that the card CVM can be altered to force the POS terminal to reveal the PIN in plaintext [30] which is acknowledged by the authors. It also does not highlight the vulnerability in which EMV cards ignore the transaction limits when the transaction is in a foreign currency [29] which is also applicable to contact Chip & PIN cards. This may be due to the particular data abstraction decisions made by the research team; which suggests that it may be a worthwhile exercise for our Z model to revisit all of the data abstraction decisions we have made to see if there may be potential avenues we have not fully explored and thereby uncover new vulnerabilities in the EMV protocol.

The approach proposed in Ouderi et al. (2013) [32] has built a formal model of the EMV protocol which they intend to use to identify vulnerabilities in the protocol and generate test cases which highlight the vulnerabilities. The research is based on the high level SysML model which captures the high level process flow and gives an easy to understand representation of the process flow.

The SmartLogic tool presented by De Konig Gans and De Ruiter (2012) [33] allows practical protocol analysis experiments to be performed upon the real-world EMV cards and POS terminals. The SmartLogic tool acts as a man-in-the-middle between an EMV card and a POS terminal during the performance of an EMV payment transaction. The tool allows the researcher to investigate the impact of a theoretical vulnerability by changing / manipulating specific commands and/or data in the EMV transaction protocol sequence and observing the resultant change in the protocol from a “correct operation” to a vulnerable state. In this way they have demonstrated the real-world impact of two known vulnerabilities which are manifest in EMV cards and POS terminals (1) the man-in-the-middle can force the EMV protocol to reveal the PIN in plaintext by altering the card capabilities (2) the

man-in-the-middle can simulate the message transmission time delays messages introduced by a relay attack proving that an EMV transaction will be completed even in a relay of 10,000km.

Choudary (2010) [34] describes the implementation of the "Smart Card Detective" tool which can manipulate the commands / data of the EMV protocol. The tool performs the man-in-the-middle protocol manipulation required to demonstrate that the attack described in Murdoch et al. (2010) [13] would work on real EMV cards and POS terminal thereby proving that the theoretical vulnerability in the EMV specification existed in actuality and could be used for fraudulent purchases.

These tools "SmartLogic" [33] and "SmartCard Detective" [34] use an approach of practical demonstration of the vulnerabilities in the EMV Specification. This is done by manipulation of the transaction protocol in real-time by a man-in-the-middle which sits between a POS terminal and an EMV card. The impact of their work is enhanced by proving that the vulnerability exists in the real world that its impact can be truly assessed.

Aarts et al. (2013) [35] presents a novel approach to analysis of the EMV protocol, in which they use a learning machine to exhaustively interrogate a real EMV card with all of the possible permutation of commands and data available in the EMV protocol. They program out a number of obviously incorrect command queries, such as incorrect Verify PIN with the wrong PIN which would block the card and halt further testing. Other than these exception cases the learning machine is free to try any valid EMV command in any sequence. EMV cards have an internal state machine, see Figure 37 which controls the order in which commands should be sent to the EMV card in a valid transaction protocol sequence. The learning machine maps the state machine of the EMV cards tested finding, by trial and error, which commands elicit what responses when the card is in a given state. This machine learning approach is completely different to our modelling approach. Despite this, the different techniques have identified a number of the same vulnerabilities in the EMV protocol via different routes:

- The MasterCard state machine enforces a set order to the command sequence of the kernel 2 protocol i.e. SELECT is followed by GET PROCESSING OPTIONS is followed by GET DATA / READ RECORDS, is followed by GENERATE AC and so on. Kernel 3 (Visa) fDDA cards are less restrictive once the SELECT command has selected the application any other command can be run.
- Kernel 2 cards default to requesting that the transaction is completed in online mode i.e. in the case where the first GENERATE AC from the POS terminal requests a Transaction Certificate (TC) offline completion of transaction, the kernel 2 card will respond with ARQC request for online completion of the transaction. At this point the specification indicates that the POS terminal must send the ARQC to the Issuing Bank and request an APRC which will indicate to that card that the issuing bank has sufficient funds and is willing to authorise the

transaction. However, both our preliminary work with our EMV emulator and the work of Aarts et al. (2013) indicate that it is possible to for the kernel 2 card to generate the TC from the second GENERATE AC command without a valid ARPC from the bank. This is achieved by setting the data fields in the GENERATE AC command to indicate that the POS terminal is offline only this triggers the card to respond with a TC without requiring the ARPC from the issuer. This creates a potential vulnerability in kernel 2 cards which can be manipulated into approving a transaction without the approval of the issuing bank.

Future work for both this research and our research includes an investigation of the implications of being able to generate a TC without issuer approval. Can that TC be utilised by a man-in-the-middle attack to approve a transaction?

The analysis work by Pasquet et al. (2008) [36] is interesting in that they set out to perform a security analysis of the contactless mobile phone payments technology "Smart Touch". The analysis method they are developing is based on Common Criteria analysis (ISO 15408). The paper proposes the use of Common Criteria analysis because it has the flexibility to deal with the complexity involved in the security analysis of a payments system which combines hardware and software elements as well as RF communications. Unfortunately at time of publication (2008) the "Smart Touch" project was still in pilot phase and was therefore not developed enough for the Common Criteria analysis to produce any analysis and/or results. On reflection the Common Criteria analysis proposed in this paper is a qualitative analysis technique relying on the experts and the community to select the criteria against which the system will be evaluated. This contrasts with our methodology and the previously discussed research [35] [32] [33] which utilise practical experiments and abstract modelling.

Ming-hui et al. (2007) proposes a number of improvements to the EMV protocol and then analyses the resultant protocol. The paper identifies a number of deficiencies in the EMV protocol which we highlight in our discussion of POS authentication, Chapter 8:

- The Merchant ID and Terminal ID are not included data which is cryptographically signed by the card to be sent to the Issuer. This potentially allows the merchant or POS terminal to act maliciously without being traceable.
- The EMV protocol does not include a mechanism for the EMV card to authenticate the identity of the POS terminal. Thereby potentially allowing malicious devices to masquerade as a POS terminal to perform unauthorised transaction with the card.

The proposed solution involves the addition of issuer public keys to the POS terminals and stricter key management at the POS terminal. Our design for POS authentication, Chapter 8, by the EMV card also required additional keys to be stored at the POS terminal. We asked for feedback on our POS authentication design from the UK Cards Association, Visa and MasterCard. We learned from this that one of the main hurdles to the implementation of such a design is that there are a large

number of legacy POS terminals which are not under the control of the Issuing banks or payment providers (i.e. MasterCard and Visa). Therefore, it is difficult to make a change to the EMV payments system which involves changing POS terminals.

4.3 Exploitable Vulnerabilities in the EMV Protocol

In this section we focus on the research papers which have identified exploitable vulnerabilities in the EMV protocol, especially those which fall into one or more of the following categories:

- the vulnerability is inherent to the protocol; which means that every EMV card and POS terminal which implements the protocol will therefore exhibit the vulnerability
- the vulnerability is exploitable; that is the vulnerability allows a transaction to be authorised under circumstances where the transaction would normally be rejected.
- the vulnerability can be demonstrated to exist in real EMV cards and POS terminal; this gives the research real-world impact
- the vulnerability compromises one of the security features in the EMV protocol; thereby allowing an attacker to authorise a transaction for which they do not have permission.

Vulnerabilities Inherent to the Protocol Specification

Murdoch et al. (2010) [13] identifies a vulnerability in the EMV payments system which allows an attacker to authorise a payment whilst entering an incorrect PIN. A man-in-the-middle device can subvert the cardholder verification process, Figure 15. Telling the POS terminal that the PIN entered by the attacker is correct, whilst telling the EMV card that this is a transaction verified by signature and therefore no PIN required. This bypasses the primary security of the EMV Chip & PIN protocol i.e. the cardholder PIN.

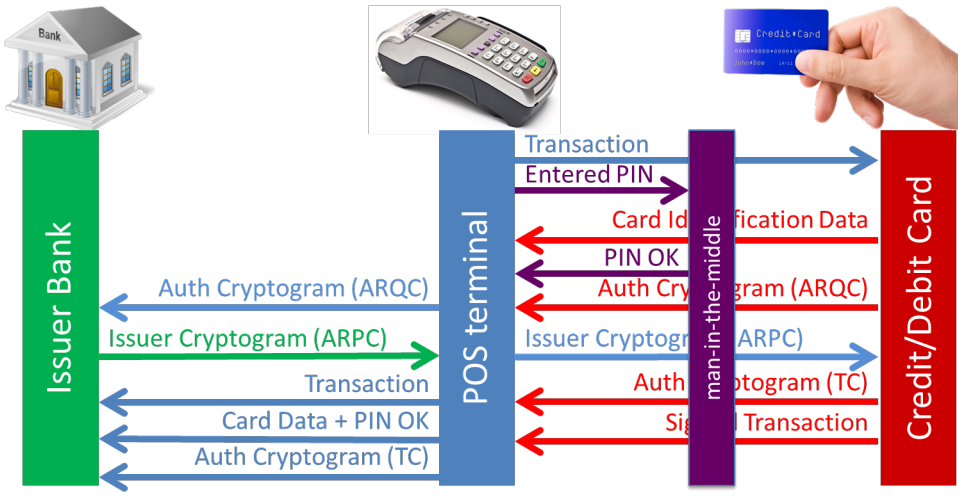


Figure 15 – Murdoch et al. (2010) protocol sequence

The research team performed practical experiments to demonstrate that the vulnerability was present in UK issued credit / debit cards and UK POS terminals. The importance of this research was

highlighted in 2012 when criminals were arrested in France, they had exploited the vulnerability to conduct 6,000 fraudulent purchases with a total value of more than €500,000 [37].

Murdoch et al. (2010) [13] uncovers critical failings in the banks transaction validation processes. The transaction data transmitted to the issuing bank includes the Terminal Verification Results (TVR) and the Issuer Application Data (IAD), which together encode the results of the cardholder verification carried out by the POS terminal and card. These data are signed by the card so the man-in-the-middle cannot alter them. However, despite this, the data required to clearly identify the fraud are split across several data fields some of which are visible to the POS terminal and others are visible to the issuing bank. This creates an ambiguity in the data which makes it difficult to detect this type of attack at either the POS or the issuer.

The EMV transaction protocol is designed to ensure that the EMV payment cards issued by many different issuing banks are accepted at any of the POS terminals / ATMs worldwide. This is a challenge as the cards, POS terminals and ATMs support multiple authorisation modes (online / offline), authorisation methods (PIN, signature, contactless) and cryptographic authentication technologies (SDA, DDA, CDA). To make any EMV card compatible with any POS / ATM, the protocol includes a negotiation at the start of the transaction to decide on authorisation mode, method and cryptography. The POS / ATM will select the most secure combination of mode, method and cryptography available to both the card and the POS / ATM.

This negotiation process is a significant weakness in the protocol. There are a number of research papers that prove it is possible for a man-in-the-middle to alter the capabilities of the card or the POS / ATM, to cause the POS / ATM to select an exploitable authentication mode, method or crypto. This type of attack is called a downgrade attack, where vulnerability is discovered in the EMV protocol and the attack must put the POS /ATM into a given mode to enable the vulnerability to be exploited. Two such attacks are Roland and Langer (2013) [26] and Barisani et al. (2011) [30].

The vulnerability discovered by Roland and Langer (2013) [26] allows the attacker to create cloned EMV contactless cards. In normal operation a cloned EMV contactless card should not be accepted, because the private key on the original EMV contactless card cannot be copied. This means that the cryptographic signature produced by the cloned card will not be validated by the POS terminal. However, the downgrade element of the attack alters the capabilities of the cards to fool the POS terminal into performing a magnetic stripe mode contactless transaction rather than the EMV mode transaction. The cryptographic protection on the magnetic stripe contactless transaction is much weaker than the combined RSA / 3-DES protection employed in EMV mode transactions. This paper demonstrates that the CVC3 code, used to authorise magnetic stripe mode transactions, can be manipulated to reduce the to number of possible values to 999. This allows a cloned card to be created with the 999 possible CVC3 responses encoded upon it.

Degabriele et al. (2012) [24] describes a theoretical partial oracle attack on the RSA cryptography used by DDA / CDA cards. This paper demonstrates that it is theoretically possible to forge the DDA / CDA cards digital signature. However, even at best, the attack would need to run 4,639 partial transactions against a card to generate the forged digital signature, making the attack impractical. Given that each transaction takes approximately 500ms, to complete the attack would require access to the card for 38 minutes.

Barisani et al. (2011) [30] was a black hat presentation at DEFCON 19 which covered a number of known issues in EMV such as the Murdoch et al. (2010) [13]. Part of which introduced a downgrade attack in which POS terminal can be convinced to reveal the PIN entered by the cardholder in plain text rather than in enciphered form, thereby allowing a man-in-the middle to record the PIN.

Vulnerabilities Caused by Omissions in the Specification

The EMV protocol specifications sometimes create the opportunity for exploitable vulnerabilities by omission; one such case is highlighted by Bond et al. (2014) [25]. The EMV specification only states that the number should be "unpredictable" it does not state how this will be achieved, it leaves this decision open to the manufacturer of the POS / ATM. The research team discovered that some POS terminals / ATMs uses unpredictable number (UN) generation techniques that were in fact very "predictable". For example, some POS terminals / ATMs simply used the time from real-time clock as the UN, some performed a hash algorithm upon the time and used that as the UN while others had a predictable sequence of UNs which reset to the beginning each time the POS / ATM is powered-up. The ability to predict the UN sequence means that the attacker can pre-generate a transaction authorisation from a genuine EMV card and use the authorisation cryptograms to create a cloned EMV card.

Anderson et al. (2005) [38] is a review of the vulnerabilities in the EMV payment system as it was in 2005, at which time the EMV cards issued in the UK were all SDA cards. As [38] highlights it was possible to create cloned SDA cards, because the authentication data supplied by the card is the same for every transaction. Cloned SDA cards have been dubbed "YES cards" as they will perform an EMV transaction with SDA static authentication; when it comes to the PIN verification step in the EMV protocol sequence the card simply replies "YES" to any PIN entered by the attacker. In the UK the process of replacing SDA cards with DDA / CDA cards, which cannot be cloned in this way, began in 2009 with the majority of cards SDA cards replaced by DDA / CDA cards by 2013.

Roland and Langer (2013) [26] and Barisani et al. (2011) [30] have demonstrated the negotiation process at the start of an EMV transaction allows the attacker to select an SDA transaction. Unfortunately, this means that despite the investment made replacing all of the SDA cards in the UK with DDA / CDA cards that the "YES card" attack may still be applicable at certain offline POS terminals.

Our work [27] demonstrates that ambiguities in the EMV contactless protocol specification [7] can lead to vulnerabilities in the real world implementation of EMV cards. Our research found that specification for kernel 2 contactless cards [39] included a statement that explicitly prohibited the use of the Verify PIN functionality in contactless mode, for "performance, usability and security reasons". We found that the specification for kernel 3 contactless cards [7] Book C-3 provided for the use of Verify PIN in contactless payments using a mobile phone, however, there was no explicit statement restricting the use of Verify PIN for contactless cards. This omission / ambiguity means that the manufacturers of contactless Visa cards have no guidance on contactless Verify PIN. Our research found that UK issued kernel 2 contactless cards blocked the Verify PIN command whilst the majority of UK issued kernel 3 cards allowed the Verify PIN command.

The vulnerability allows unauthorised access to the Verify PIN functionality of EMV contactless cards and potentially allows an attacker to compromise the cardholder's PIN whilst the card is still in their wallet. We built a scenario where the attacker would guess the cardholder's PIN whilst the cardholder was presenting his/her wallet to their building entry system each day. We also identified that the vulnerability would allow an attacker to temporarily block any contactless Visa cards in the wallet by making 3 deliberately incorrect PIN attempts on each card. This would be an effective denial of service attack on the cardholder causing them inconvenience and embarrassment the next time they attempted to use their cards.

The abstract Z model in our analysis methodology, Chapter 5, allows us to identify omissions in the EMV protocol specification. One such omission is discussed in "Harvesting High Value Foreign Currency Transactions from EMV Contactless Credit Cards without the PIN", Chapter 6. This is an exploitable vulnerability which compromises the security features of the EMV protocol. Contactless payments are intended low value transactions, under £20 (May 2015) in the UK. Above £20 the cardholder must enter their PIN giving additional security for higher value transaction. The vulnerability allows an attacker to authorise transactions significantly greater than £20 without the cardholder's PIN. The vulnerability is present on the contactless interface of the card, this potentially allows an attacker to exploit the weakness whilst the contactless card is still in the cardholder's wallet. In order to illustrate the "real-world" impact of the vulnerabilities identified by our research [29] [27] we develop practical demonstrations using Android mobile phones. This shows that the vulnerability exists in actual EMV cards and POS terminals and that it can be exploited using off the shelf equipment available to the fraudsters.

4.4 Exploitable Vulnerabilities in EMV Contactless Payment Technology

There are a number of known vulnerabilities in the underlying technologies which support EMV contactless payments, these can be split into the following categories:

- contactless transaction relay
- eavesdropping contactless payments
- extending the effective range of NFC

These categories of vulnerability have one thing in common, they take advantage of the underlying contactless / NFC technology they are very difficult to guard against and or prevent.

Contactless Transaction Relay

EMV contactless payments make the security assumption that the cardholder has authorised the transaction because the EMV card was placed on the POS terminal. This is based upon the limited range of the underlying ISO-14443 wireless technology [40] which has a maximum effective range of only 10 cm. Relay attacks break this security assumption allowing transaction to be completed whilst the payment card is potentially many kilometres away from the POS terminal.

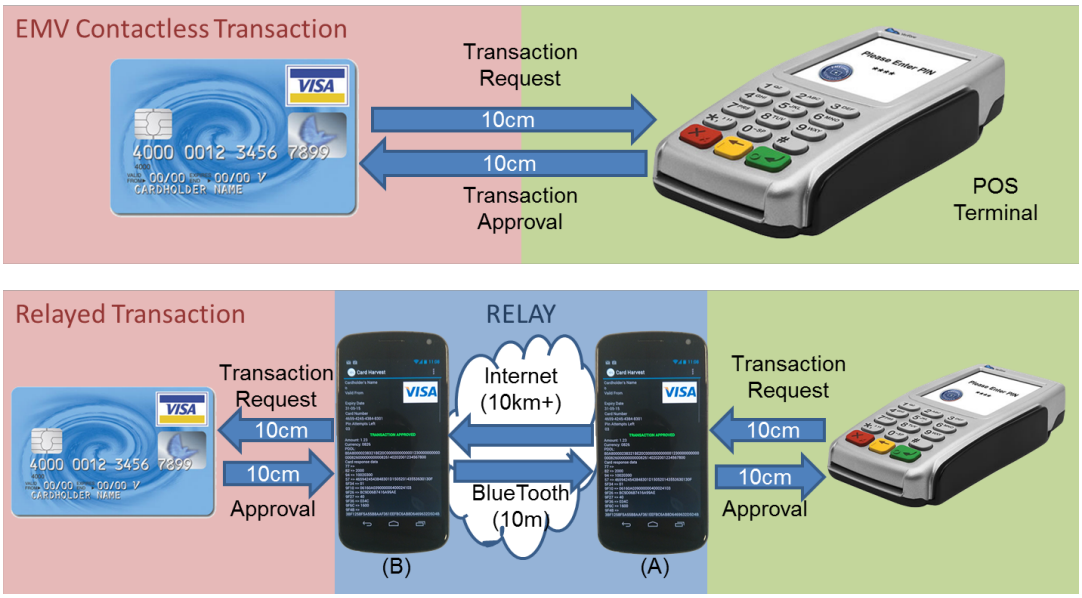


Figure 16 - EMV Contactless Transaction Relay

Figure 16 illustrates how a Contactless Relay attack is carried out. Relay attacks are relatively simple, which makes them easy to implement and very hard to prevent. The attacker has two NFC enabled mobile devices. The "proxy" device (A) communicates with the POS terminal. The "mole" device (B) communicates, with the victim's genuine EMV card.

(A) receives data requests from the POS terminal. (A) sends the requests to the (B), which passes the request to the genuine card. (B) takes the genuine responses from the card and relays them to (A). (A) in turn passes the genuine responses to the POS terminal.

There is no processing of the data involved in the relay. (A) and (B) simply relay messages from the POS terminal to the card and responses from the card to the POS. Both the card and the POS terminal are unaware that they are not 10cm apart. The relay between the "proxy" (A) and the "mole" (B) can

be either Bluetooth or can be via the mobile phones connection the Internet. Bluetooth is limited in range 10 metres but has the advantage of being a dedicated connection with guaranteed communication speed. Using an internet connection enables the relay to take place over much larger distances (many kilometres), however, the communication speed is not guaranteed which may cause the relayed transaction to drop out.

Francis et al. (2009) [41] demonstrates that NFC enabled smartphones can provide a perfect platform for both the proxy (A) and the mole (B) as they combine NFC and go anywhere internet communications. They are also innocuous as smartphones are now starting to have EMV compatible payment applications installed on them. A recent publication by the team at RHUL demonstrates that relay attacks are a very real threat to EMV payments; Francis et al. (2012) [42]. Publications by Cambridge University and Tel Aviv University reinforce this research. Drimer and Murdoch (2007) [43] and Kfir and Wool (2005) [44].

Contactless payments cards are "passive", they can be activated to make a payment by any ISO-14443 reader that comes within range. Mobile phone payment devices are more secure in that the payment applications, such as Google Wallet, require the user to unlock the application before a payment is made. This means that unlike the "passive" cards, the mobile phone payment device cannot be attacked whilst still in the cardholder's wallet.

However, once the mobile payment device is active the mobile phone payment device performs exactly the same contactless transaction protocol as the contactless cards. This means that the mobile phone payment can be relayed in the same ways as a contactless card payment. This is demonstrated in the research Roland and Scharinger (2013) [45] and Roland et al. (2012) [46]. This research demonstrates a novel vector of attack, a malicious app resident on the Google Wallet mobile phone acts as the relay "mole" (B). The malicious app activates the Google Wallet payment app and sends a payment to a waiting "proxy" (A). This attack vector means that the attacker does not have to get physically close to the Google Wallet device as is the case in the classic relay scenario Figure 16. The attack vector makes this attack scenario particularly dangerous as the attacker can take payments from any mobile phone that installs the malicious app. The app then transmits these payments to any location with internet access where the payment is collected by the "proxy" (B) and a collaborating, possibly malicious, merchant with a POS terminal.

This research shows that mobile phone payments are just as vulnerable to relay attack as EMV cards, despite having the added security of a PIN code unlock. Further to this our testing of the Orange QuickTap mobile phone payment app found that once the payment app had been activated to make a payment the payment functionality would remain active for several minutes after the screen had gone blank. This would allow a short window of opportunity for a relay attack subsequent to a legitimate payment having been made with the QuickTap phone.

Eavesdropping, Skimming and Extended Range Reading

Contactless payments utilise the ISO-14443 wireless communications standard [40], which is an open standard used in many different contactless applications on smartcard and mobile devices. Use of this common standard leaves contactless payments vulnerable to data hijacking attacks such as eavesdropping, skimming and extended range reading. The data gathered by these attacks include the 16 digit card number (PAN) and the card expiry date, which research shows is sufficient to create a new account on Amazon.com and make online purchases [47] [48]. This is due to the minimal security checks on some websites which do not enforce a full check on all of the card-not-present security fields recommended by EMV, i.e. the PAN, expiry date, CVV2, cardholder name and cardholder address. Therefore despite the cryptographic security that prevents cloning of EMV cards based on the data obtained through contactless eavesdropping, skimming and extended range reading; the data collected are still useful in performing card-not-present attacks.

Eavesdropping

A number of research projects have looked into the practicalities of eavesdropping the ISO-14443 wireless communications. These projects show that it is possible to eavesdrop the data from a contactless payment at a distance of 1 metre. The research does prove that eavesdropping produces exploitable data, thereby making the contactless EMV cards vulnerable to attack. However, the research also shows that the equipment required to perform contactless eavesdropping is very specialised requiring a great deal of electronics expertise to build. For instance Diakos et al. (2015) [49] presents excellent research which builds the eavesdropping equipment into everyday objects such as a shopping trolley. However, as the research also shows, the RF receiver and the signal processing equipment required are complex and would require a great deal of work to make the equipment portable enough to be used in real-word attack scenario.

This would make eavesdropping a much less attractive method of collecting credit card data when compared with skimming attacks using an NFC enabled mobile phone. Research by Francis et al. [41] [42] and our research [47] [29] show that skimming attack can be performed using off-the-shelf Android mobile phones which are very portable and discreet.

Hancke et al. (2011) [50] makes a comparison between eavesdropping and skimming attacks using the same equipment. The result of the comparison between the eavesdropping and skimming concludes that eavesdropping has the potential to read from a greater distance, however, the skimming provides more reliable data reading. With eavesdropping being more susceptible to atmospherics, environmental conditions and RF interference.

Skimming

The popularity of NFC enabled Android mobile phones provides a perfect attack platform for contactless skimming as demonstrated in [42] [47] [29]. However, that is not the only potential attack vector, we developed an attack platform that masquerades as a NFC door reader [27]. The door reader accesses all of the cards in a victim's wallet before activating the door opener. Our multiple card reader software utilises the standard anti-collision functionality present in the ISO 14443 standards [40] (part 3). One possible attack scenario could have our multiple card reader deployed as an Oyster card turnstile or ITSO card turnstile, in such a scenario the attack could capture the details of many thousands contactless payment cards.

Our experimental research [51] presented a practical skimming attack costing only £40, which would allow a shop assistant to collect credit card details to make fraudulent online purchases.

Extended Range Contactless Reading

The maximum practical communication range of EMV contactless payments cards is approximately 5cm. EMV uses the restricted communications range of ISO 14443 as a design security feature. The cardholder authorises the payment by tapping their card on the POS terminal, the assumption being that the cardholder must be present at the merchant location to authorise the payment.

There has been significant research into the extending the read range of contactless payment cards. Kirshenbaum and Wool (2006) [52] demonstrates that ISO 14443 cards can be read at a distance of 30cm which is 6 times the design distance. The experiments show that to increase the effective reading range of ISO-14443 cards that reader must increase the transmission power from 200mW to 4 Watts and increases the antenna size from 5cm diameter to 50cm diameter.

Hancke et al. (2011) [50] introduces an interesting concept, it utilises two separate antennas to extend the reading range. A standard ISO 14443 reader uses a single antenna to power the card, transmit data to the card and to receive the card responses. The two antenna approach uses one antenna to power and transmit, it uses the second antenna to receive the card responses. Using a second receiving antenna allows the attack to increase the reading range of ISO 14443 whilst using less signal power and smaller antenna diameters.

One of the attack scenarios explored in Oren et al. (2013) [53] is a "mafia fraud attack" scenario. The POS terminal ("ghost") which is dedicated to receiving fraudulent transactions. An extended range contactless reader ("leech") is used to capture transactions from passing victims at a range of 1 meter whilst the contactless payment card is still in the victim's wallet. The "ghost" and the "leach" are connected by the relay allowing them to be many kilometres apart.

4.5 Review of Known Vulnerabilities in the EMV Payment System

Markantonakis et al. (2009) [54] provides a good overview of the published research concerning practical attacks on EMV cards. It places EMV cards in the context of all of the different applications using similar smartcard technology. It identifies skimming, eavesdropping, relays and extended range reading as being the primary threats against contactless payment cards.

Anderson et al. (2005) [38] reviews the known vulnerabilities in EMV. It explores the real-world impact of the vulnerabilities in terms of:

- monetary value of the fraud attributed to an attacker exploiting the vulnerability
- the indirect impacts of being a victim of fraud upon the cardholders and merchants
- the cross over impacts on other areas of the payments system e.g. magnetic stripe data being used to commit card-not present fraud.
- the EMV fraud liability shift from the banks to the merchants and cardholders. In the UK prior to the introduction of EMV the banks were liable for the cost of all card fraud in the UK. Post EMV the merchant or cardholder can be held responsible e.g. if the correct PIN is used the bank will assume that the customer was negligent with their PIN and is therefore liable for the cost of the fraud.

The UK and Europe migrated from magnetic stripe card payments to EMV Chip & PIN based payment technology in 2004. The United States is currently (May 2015) in the process of migration from magnetic stripe to EMV Chip & PIN. US banks have started issuing EMV payment cards and US retailers, such as Wal-Mart, have commenced a rolling out of EMV compatible POS terminals.

The card payment scheme providers are using the move to EMV to bring about a liability shift in the United States [1] [3] [2]. This has raised discussion in the US news media that customers and retailers will have greater exposure to fraud losses than was the case for the magnetic stripe payments system.

Anderson (2007) [55] highlights that, of all of the known vulnerabilities of contactless payment cards it is the ability to communicate with the card without the knowledge of the cardholder that is the fundamental flaw in the contactless design.

4.6 Contribution of Literature Review to this PhD Research

This literature review has focused on four areas of academic research related to the security of the EMV contactless protocol; structured / formal analysis of the EMV protocol, exploitable vulnerabilities in the EMV protocol, exploitable vulnerabilities in contactless technology and review of known vulnerabilities in the EMV payment system. The research in the literature review influenced the research presented in this PhD thesis as follows.

Structured / formal analysis of the EMV protocol

Research relating to structured / formal analysis of the EMV protocol followed two approaches; building a formal model of the EMV protocol or building tools which analyse EMV protocol messages of actual transactions in real-time. Formal analysis can identify errors in the specification which are outsides the bounds of normal operation and therefore are not picked up in testing. The protocol analysis tools are good at identifying vulnerabilities in the real-world operation of the protocol and injecting errors to demonstrate the weaknesses.

De Ruiter and Poll (2011) [28, 31] is the most comprehensive formal analysis. However, the analysis only identified a number of known vulnerabilities in the protocol and did not predict any new vulnerabilities, which have since been identified by other research.

There were a number of very good protocol analysis tools in the academic research by De Koning Gans and De Ruiter (2012), Choudary (2010) and Pasquet et al. (2008), which have been used to identify vulnerabilities in the EMV protocol. The drawback of the analysis tools is that they examine the protocol as it is implemented by the software developers, which is their interpretation of the meaning of the original specification. This makes it difficult to predict the edge-case vulnerabilities generated by fundamental errors / omissions in the specification.

Our approach was to develop a methodology, Chapter 5, which combined both analysis techniques. As the literature shows the strength of formal in predicting outlying edge cases mitigates the weakness of the tool based approach that it is restricted to testing what has been implemented by a 3rd party. The strengths of analysis tools reside in the analysis of real-word operation of the protocol, thereby helping us refine the formal model so that it closely follows the real-world operation of the protocol and overcomes some of the ambiguities in the EMV specifications.

Our approach is the first in the literature to successfully combine the formal and tool based approaches and to use the resulting methodology to identify previously undocumented vulnerabilities in the EMV protocol.

Exploitable vulnerabilities in the EMV protocol

Our methodology was also influenced by the research into exploitable vulnerabilities in the EMV protocol. At the outset of my, PhD researchers at Cambridge University (Mike Bond and Ross Anderson) encouraged the use of practical experiments in my research. This allowed me to demonstrate that the vulnerabilities the methodology highlighted in the protocol were exploitable in the real world, and thereby increase the impact of the research. This approach has proved very successful in our papers Emms et al. (2013) and Emms et al. (2014) which were accepted at FC 2013 and CCS 2014 respectively. The influential papers in the area Murdoch et al. (2010), Roland and Langer (2013) and Bond et al. (2014) use this approach. Using practical demonstration reinforcing

the impact of the discovery of a vulnerability in the protocol that could otherwise be easily dismissed as a minor technical issue.

Exploitable vulnerabilities in the EMV contactless technology

Research into the exploitable vulnerabilities in the EMV contactless technology, have both influenced and confirmed the technology choices made in our experimental work. Work by Hancke (2011), Roland and Scharinger (2013), Roland et al. (2012) demonstrated the use of NFC enabled smartphones as a practical attack platform against EMV contactless payments. Their work was concurrent with our work and helped to confirm our direction.

Work such as Francis et al. (2012) on relay attacks provides indirect supporting evidence for my research. The primary focus of my research is analysis of the protocol, a relay works without understanding the content of the protocol messages. However, in some of our research we build upon the mobile phone to mobile phone relay, by inserting a man-in-the-middle which alters the protocol to exploit a specific protocol vulnerability.

Work by Kfir and Wool (2005) and Diakos et al. (2015) on extending the range of NFC comprehensively explores extended range reading and eavesdropping contactless payments. This allowed me to decide that to develop the hardware required for long range NFC reading would divert time and effort from analysis of the protocol which was my primary research goal, without adding to the scientific knowledge in this area.

Review of known vulnerabilities in the EMV payment system.

Papers in this category look at the system wide impacts of the individual vulnerabilities in the EMV payment system. This gives context to our research and has helped me to more fully understand the impact of the vulnerabilities identified by my research and assisting me to convey this message to a non-academic audience; the general public, law enforcement and the payment industry.

4.7 Conclusion

There are several leading academic research teams actively analysing the security of the EMV transaction protocol and researching potentially exploitable vulnerabilities in the EMV protocol. These researchers have adopted various approaches including formal modelling and practical experiments on real EMV cards, POS terminals / ATMs. In this literature review I have established a link between the existing academic research and the three areas of weaknesses in contactless payment identified in Chapter 1:

- the wireless interface makes EMV contactless payments vulnerable to several new categories of attack (i.e. skimming, eavesdropping and relay) which were not present in Chip & PIN.

- contactless payments do not require PIN verification; consequently many of the attacks described in the literature review are only possible because there is no cardholder interaction required to complete the transaction.
- the introduction of contactless payments increases the complexity of the EMV authentication process, Figure 9. This exacerbates a pre-existing problem identified in the literature review, whereby the complexity of the EMV authentication process facilitates many of the attacks, in particular man-in-the-middle and downgrade attacks.

The EMV contactless protocol is just one part of the EMV payment system which also includes card-not-present payments, magnetic stripe compatibility mode payments and Chip & PIN payments. This means the security of the EMV protocol and the contactless technology that supports it must be analysed in the wider context of the EMV payment system. There are several cases where a vulnerability in the contactless transaction protocol / technology does not affect the security of contactless payments but does affect one of the other EMV payment methods.

The literature review supports the assertion made in this PhD thesis that the EMV payment system is fundamentally weakened by the EMV philosophy of providing support for all previous card payment technologies including magnetic stripe, SDA and printed plaintext data in the case of card-not-present payments. It is apparent that despite the investment made in improving the security technology of EMV payments cards, the requirement for backward compatibility reduces the security to the least secure technology supported by the system.

Chapter 5. Analysis Methodology

This chapter describes the methodology, Figure 17, developed as part of this PhD research, to analyse the security of the EMV contactless protocol. Our analysis methodology combines the strengths of formal modelling with a protocol emulator which provides real time analysis of the protocol as it is implemented in the EMV payment system. The two models complement each other, the formal model predicts potential vulnerabilities in the protocol based on the EMV specification, while the protocol emulator allows us to evaluate the real-world impact of these potential vulnerabilities on the EMV protocol.

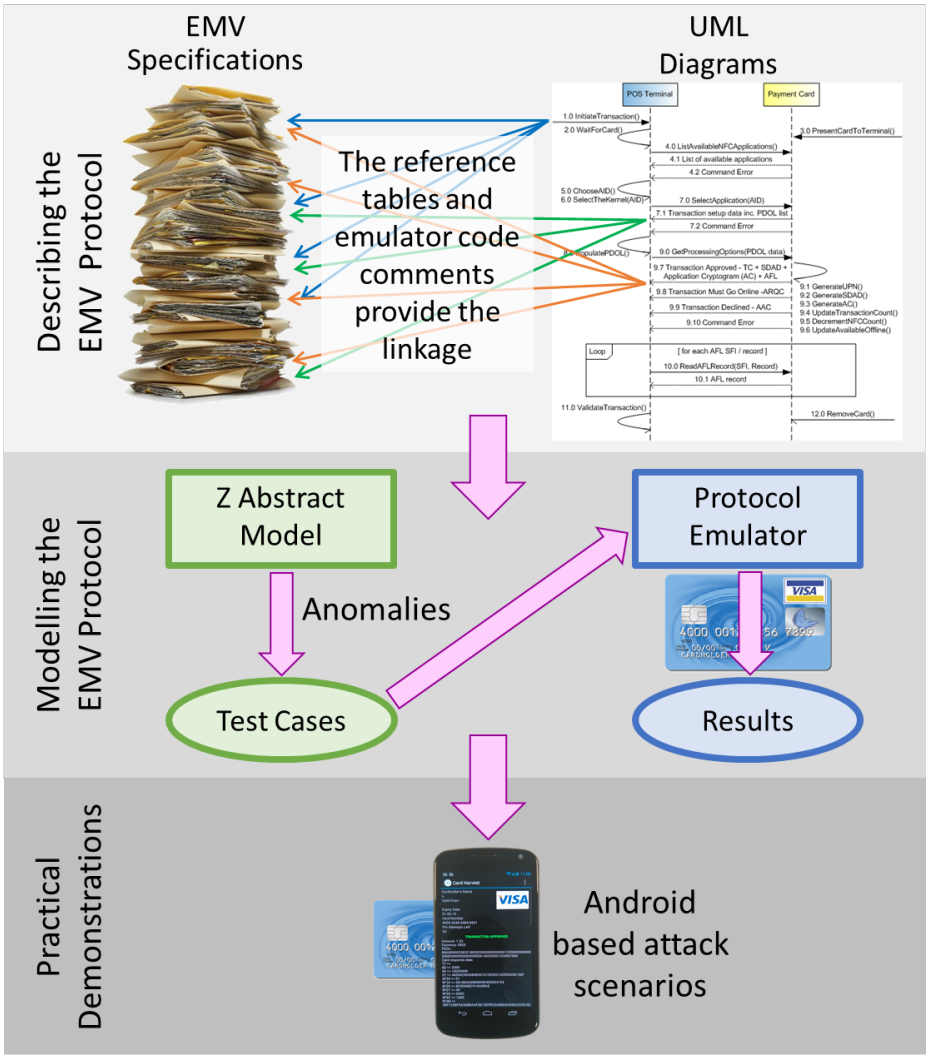


Figure 17 - Overview of Analysis Methodology

The analysis methodology, Figure 17, combines three elements:

- Describing the EMV protocol; uses UML diagrams and reference tables to describe the EMV protocol and provide a documented linkage back to the EMV Specifications [6] [7].
- Modelling the EMV protocol; our methodology combines a Z abstract model with a protocol emulator. The *Protocol Emulator* allows us to execute EMV protocol sequences with real EMV cards, thereby allowing us to evaluate the impact of vulnerabilities identified in the EMV specification. The *Z Abstract Model* highlights potential vulnerabilities in the EMV specification and generates test cases for the protocol emulator.
- Practical demonstrations; we have developed a number of *Android based attack scenarios* which show that the vulnerabilities identified by our analysis methodology, can be exploited using off-the-shelf equipment. This demonstrates the real-world impact of the vulnerability

The objective of the analysis methodology, Figure 17, is to evaluate the security of the EMV transaction protocol. The evaluation process is carried out using the protocol emulator together with the Z abstract model. Once the modelling has identified a vulnerability in the protocol our methodology provides a link to the location(s) in the EMV specification which have generated the vulnerability. This is achieved by a combination of the UML diagrams, the reference tables and structured comments in the protocol emulator code. It provides documented evidence that the vulnerability is attributable to an inherent flaw in the specification rather than being simply an implementation error by a manufacturer of a particular EMV card.

Our practical demonstrations are designed to communicate the results of our research to a non-technical audience. This increases the impact of our work making our security message accessible to members of the public who are carrying EMV contactless cards.

5.1 Describing the EMV protocol

The EMV transaction protocol specifications [6] [7] are lengthy and complex, consisting of 14 books and 2392 pages. Contactless payments added greatly to that complexity comprising 10 of the 14 books and 1645 pages. A great deal of this complexity derives from the inclusion of seven different contactless protocol sequences; one for each of the payment scheme operators, MasterCard, Visa, American Express, JCB, Diners, Discover and UnionPay, whereas, Chip & PIN has a single protocol sequence for all card types.

A major challenge for the analysis of the EMV protocol was to condense the information in the EMV specifications into a manageable format, which was easier to work with (than the 2392 pages of the EMV specification) but which retained the linkage between the original specification text and the models we have built based on the specifications. The analysis methodology uses a combination of a UML protocol sequence diagram with associated reference table to achieve this.

5.2 UML Diagrams

The role of the UML protocol sequence diagrams is to collate information from multiple sources in the EMV specification to create a single description of the transaction protocol sequence (kernel). There are eight transaction protocol sequences (kernels) in the EMV specification, one for contact transactions and seven for contactless transactions. There is a single UML diagram for each of the eight kernels. Figure 18 shows the UML diagram for kernel 3 fDDA contactless protocol sequence.

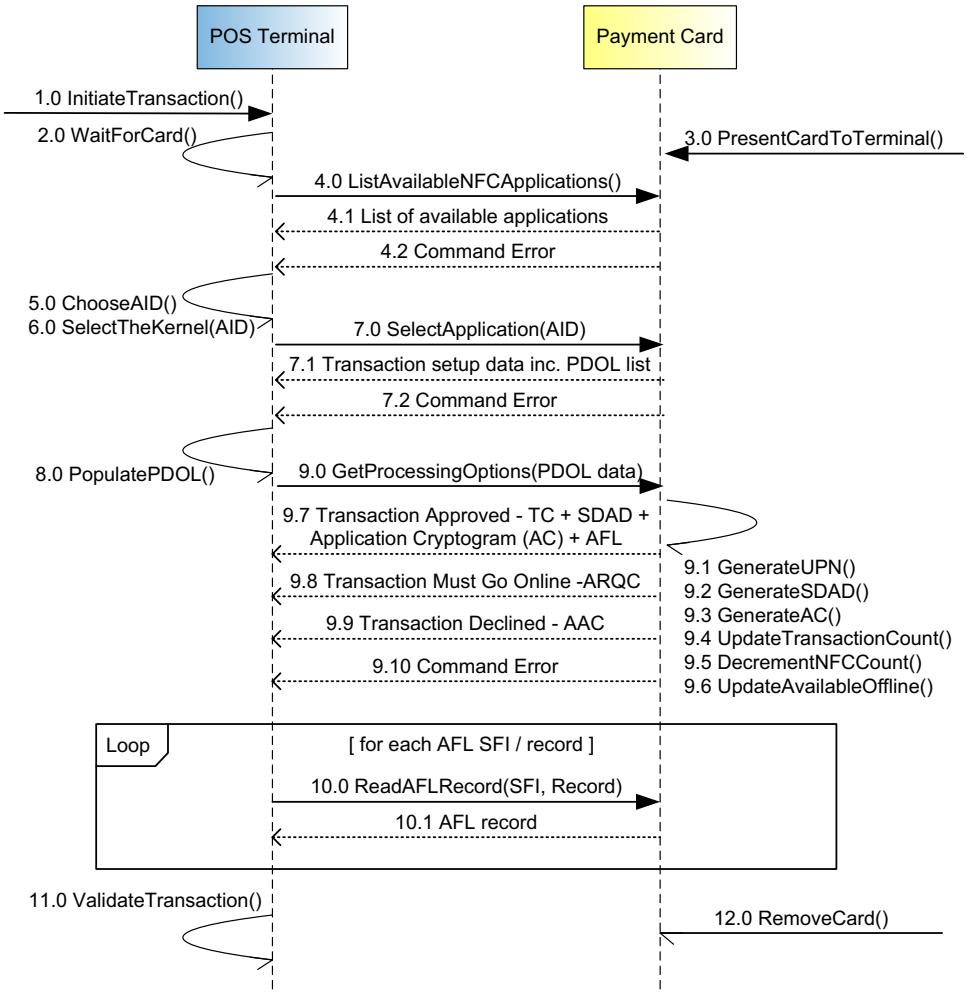


Figure 18 – Kernel 3 fDDA Contactless Protocol Sequence

Each message, response and function is numbered to provide an easy to follow documented link with the reference table and the java code of the protocol emulator. The complete transaction protocol sequence consists of a number of command messages sent by the POS terminal and responses returned by the payment card. The numbering scheme reflects this; the initial commands are given a new whole number (e.g. 7.0) all related responses are given numbers in that range (e.g. 7.1, 7.2, 7.3 etc). Some actions are initiated by an external user such as 1.0 InitiateTransaction() and 3.0 PresentCardToTerminal(). Other actions are initiated and completed internally within either the

payment card or the POS terminal such as 5.0 ChooseAID() and 9.1 GenerateUPN(), these are usually actions which create or select the data required by the next message or response.

5.3 Reference Tables

Each UML protocol sequence diagram is accompanied by a reference table. The table provides a detailed description of each message and each response in the UML protocol sequence diagram and a list of references in the original EMV specification from which the UML and descriptive text are derived.

The following example of an entry in the reference table, Figure 19, reflects the numbering scheme of the UML diagram, Figure 18, to create the link between UML diagram and the table. Figure 19 shows a single table entry corresponding to *9.7 Transaction Approved - TC, SDAD, Application Cryptogram (AC), AFL* in the UML diagram. A full example of the linkage between the UML diagrams, reference tables, protocol emulator code and the Z abstract model code are given in Appendix A.

The descriptive text of the table provides an overview of the functionality of each message / response in the protocol sequence. The references consist of the EMV book with version (`EMV v2.2 Book C-3`), page number (`p50`) and section number / title (`5.4.3 Determine the Card Transaction Disposition`). This provides implementation level details required for the coding of the protocol emulator and the creation of the abstract model. The same references appear in the protocol emulator code and provide a documented linkage between a vulnerability observable using the emulator and its origin in the EMV specification.

Descriptive Text	References
←9.7 Transaction Approved - TC, SDAD, Application Cryptogram (AC), AFL If the card approves the completion of the transaction in offline mode, it will return Transaction Cryptogram (TC) in the Cryptogram Information Data (CID). The card also returns all of the data elements required by the terminal to complete the transaction: Signed Dynamic Application Data (SDAD) used by the terminal to verify that the card has approved the same transaction that the terminal sent. Application Cryptogram (AC) used in the completion of the transaction with the Bank to validate that a valid card completed the transaction. Application File Locator (AFL) contains the location in the card's file structure where the terminal can read the data elements required to complete the transaction.	EMV v2.2 Book C-3 - p50 5.4.3 Determine the Card Transaction Disposition EMV v2.2 Book C-3 - p43 5.2.2.2 GPO Response SW1 SW2 EMV v2.2 Book C-3 - p46 5.2.2.3 Contactless Path Determination EMV v2.2 Book C-3 - p97 A.2 Data Elements by Name EMVv2.2 Book C-3 - p127 Annex C Fast Dynamic Data Authentication EMV v2.2 Book C-3 - p128 C.1 Dynamic Signature Verification EMV v4.3 Book 3 - p59 6.5.8 Get Processing Options APDUs EMV v4.3 Book 3 - p60 6.5.8.4 Data Field Returned in the Response

Figure 19 - Section of Reference Table

5.4 Modelling the EMV Protocol

The analysis methodology comprises two models, the protocol emulator and the Z abstract model, which are used in concert to identify vulnerabilities in the protocol sequence and to analyse the impact (exploitability) of the vulnerabilities.

5.5 Protocol Emulator

The protocol emulator is at the centre of our analysis methodology, it provides a functional model of the EMV contactless protocol sequences. The protocol emulator consists of a PC application which emulates a POS terminal and an Android mobile phone app which emulates an EMV contactless card. Figure 20 shows the POS Terminal emulator and the card emulator performing a transaction together.

Figure 20 – POS Terminal Emulation with EMV Card Emulator (Android phone)

Having software emulations of both sides of the EMV contactless protocol sequence, the POS terminal and the card, allows us to:

- use the POS terminal emulator to execute the protocol sequence with real EMV cards
- run fully emulated protocol sequences between the protocol emulator and the card emulator. We generated self signed public / private key pairs for the card emulator, with a matching self signed CA public key on the protocol emulator. This allows the card emulator to generate a SDAD signature which can be validated by the protocol emulator.
- the EMV card emulator can execute partial protocol sequences with a real POS terminal. However, transaction authorisation will always fail, because the emulator does not have the bank issued private keys to generate a SDAD that will validate in a real POS terminal.

The POS terminal emulator is the primary element of the protocol emulator. The POS terminal emulator encapsulates the modelling of the EMV protocol sequences, the linkage to the UML diagrams and EMV specifications, as well as providing a fully functioning POS terminal.

The Android EMV card emulator provides the functionality of the card side of the protocol sequence and records log traces from the card's viewpoint.

5.5.1 POS Terminal Emulator

The POS emulator allows us to run EMV protocol sequences with erroneous and edge case data, it also allows us to insert erroneous commands into the protocol sequence. This gives us the ability to simulate different protocol scenarios based on our analysis of the EMV protocol. The structured code comments in the protocol emulator allow us to link any erroneous behaviour observed by the emulator back to the EMV specification.

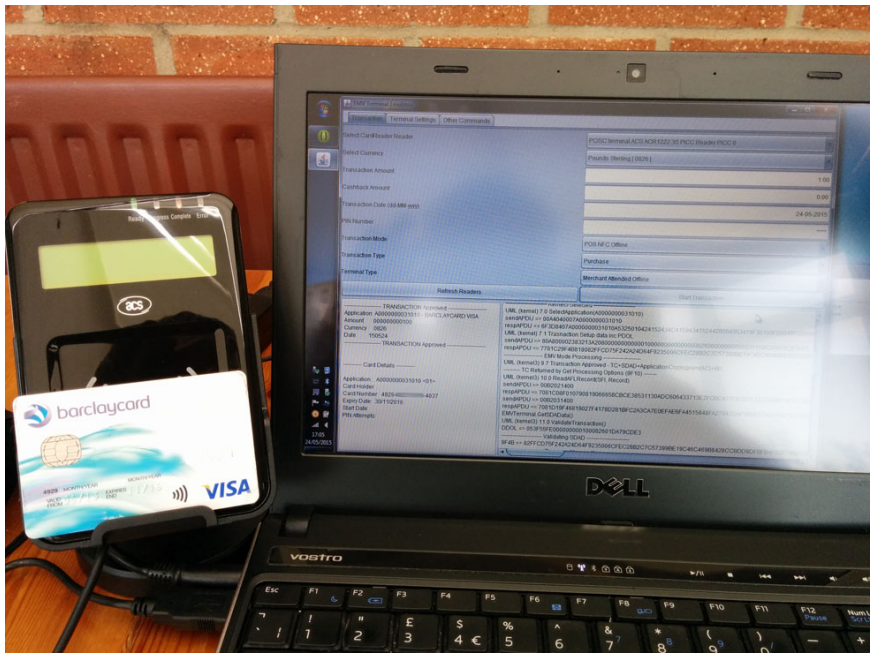


Figure 21 - POS Terminal Emulator Performing Kernel 3 Protocol Sequence

Figure 21 shows the POS terminal emulator running the Kernel 3 fDDA contactless protocol sequence, Figure 18, with a real EMV card (my Barclaycard Visa).

Figure 22 is a screen shot of the POS terminal emulator showing the execution of the steps in the protocol sequence with my Barclaycard.

The top part of the screen shows the transaction parameters that will be used in the protocol sequence i.e. this is a purchase transaction, the currency is UK pounds, the transaction amount is £1.00 and the date is 24-05-2015. Transaction Mode sets the terminal capabilities; in this case POS NFC Offline. Setting the terminal to offline means that the card will not request that the transaction is authorised by the Issuing bank using the ARQC authorisation cryptogram, the card will simply produce the TC transaction certificate to complete the transaction, Figure 14.

The terminal type merchant attended denotes that this is a POS terminal in a shop / restaurant with a member of staff operating the POS terminal (not an unattended petrol pump or vending machine).

The bottom part of the screen contains the results of the running protocol sequence. On the left is the result of the protocol sequence, in this case an approved contactless transaction for £1.00, currency 0826 (UK pounds) and dated 24-05-2015 (reversed 150524). Below the approved transaction shows the card details, that the transaction was approved by my Barclaycard.

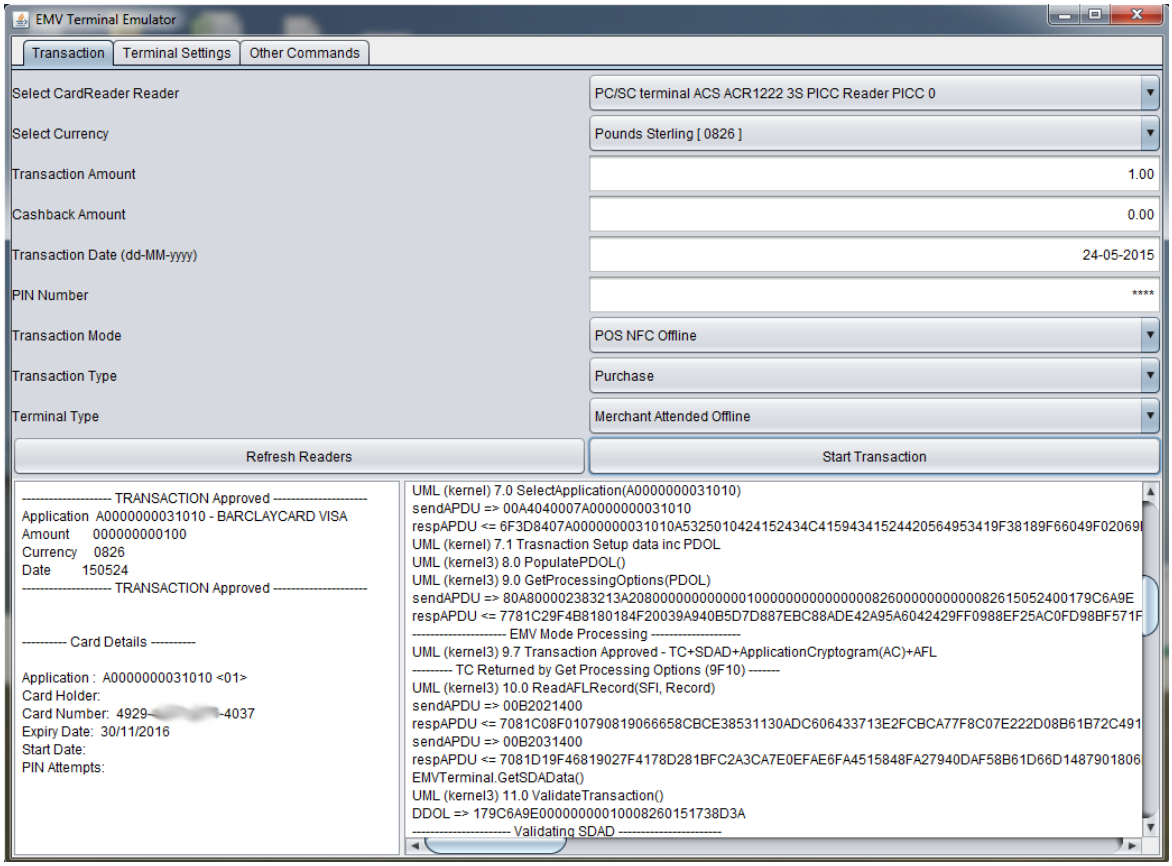


Figure 22 – POS Emulator fDDA Transaction with Log Trace

The window on the right contains the log trace from the transaction protocol sequence. In this case it shows the section of the protocol which corresponds to the UML diagram, Figure 18 step 7.0 SelectApplication() to step 11.0 ValidateTransaction().

The log traces of the protocol emulator provide

- a list of the protocol sequence steps executed by the emulator. This links the results of the emulator experiments with the EMV specifications via UML diagram numbering scheme, e,g. *UML (kernel 3) 8.0 PopulatePDOL()*
- details of the messages and data sent to the card by the POS terminal i.e. *sendAPDU =>*
- details of the card's responses to the POS messages *respAPDU <=*
- decision point milestones, to make it easier to follow the process flow e.g. *---TC Returned by Get Processing Options (9F10) ---*

- details of dynamically generated data structures such as the *DDOL => 179C6A9E...* which will help ease the process of interpreting the log trace

A full trace log for this protocol sequence can be found in Appendix A.4.

Figure 22 shows the UML entries in the trace log which are used to provide a linkage between the running code which is performing the protocols sequence to the UML diagrams and reference tables. This creates a documented link from the error occurring in the trace logs => UML diagrams => reference tables => EMV specification (book, section number and page).

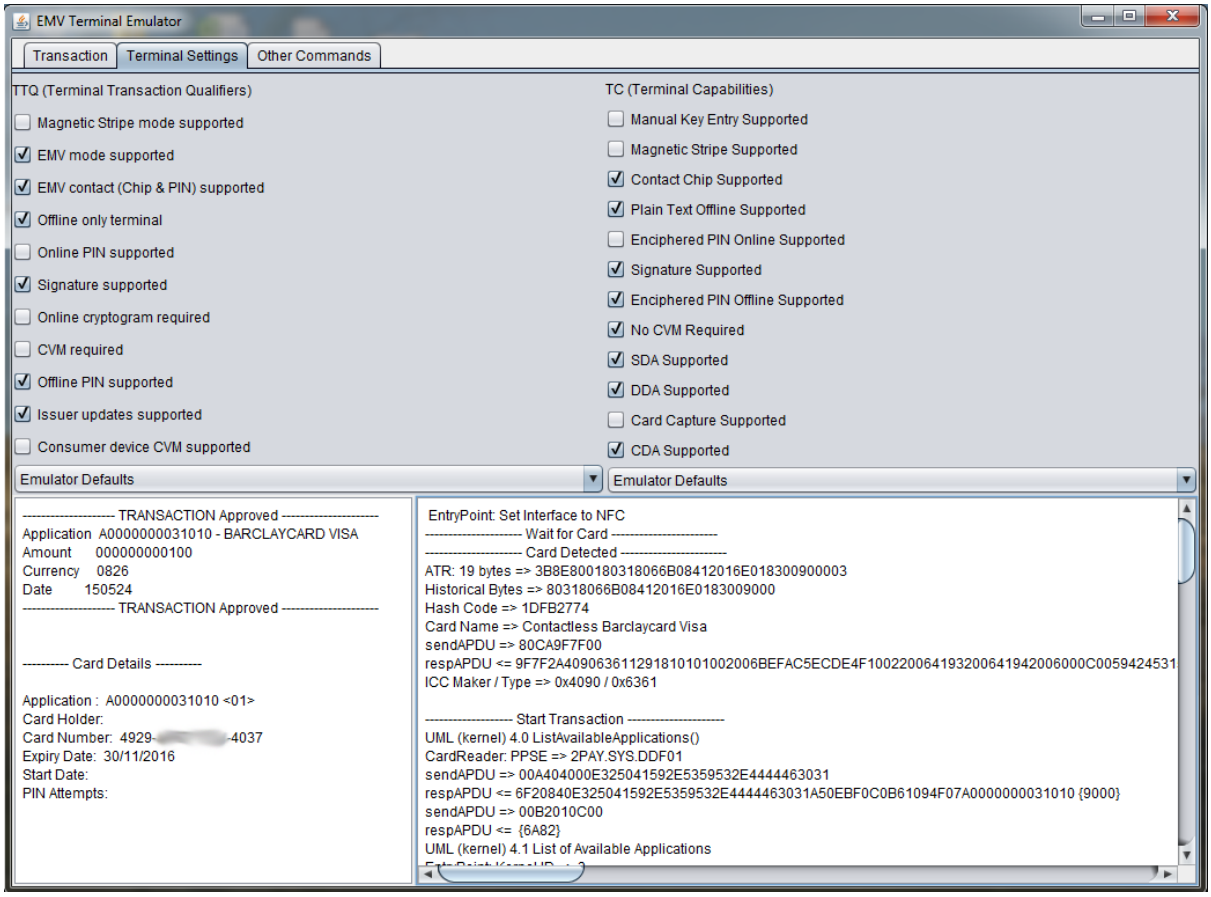


Figure 23 - POS Terminal Emulator Parameter Settings

Figure 23 shows the parameter settings that set up the protocol sequence. This allows the POS terminal emulator to run a protocol sequence that targets a particular security vulnerability. For instance by setting the EMV parameter switch of an Offline only terminal, we can force the EMV card not to request Issuer approval using the ARQC. We can also switch off the more secure features of EMV such as enciphered PIN, forcing the card to perform plaintext PIN.

This exploits the process by which EMV selects which security features will be utilised in this protocol sequence, Figure 9. It forces the EMV card to perform the security options which allow a vulnerability to be exploited; the attack outlined by Roland and Langer (2013) [26] does exactly this.

It downgrades the POS terminal from EMV mode to Magnetic Stripe mode, thus bypassing the stronger cryptographic protection present in EMV mode contactless transactions.

5.5.2 Executing Bespoke Protocol Sequences

The POS Terminal Emulator also has the capability to execute protocol sequences such as the contactless PIN verify sequence, Figure 35, described in section 7.2.1. This functionality is used to specifically target vulnerabilities that are present in the EMV specifications but which do not fall within the usual format of an EMV protocol sequence as was the case for the contactless PIN verify vulnerability. The POS terminal emulator runs the bespoke protocol sequence and records the responses of real EMV cards for evidence of the presence of the vulnerability.

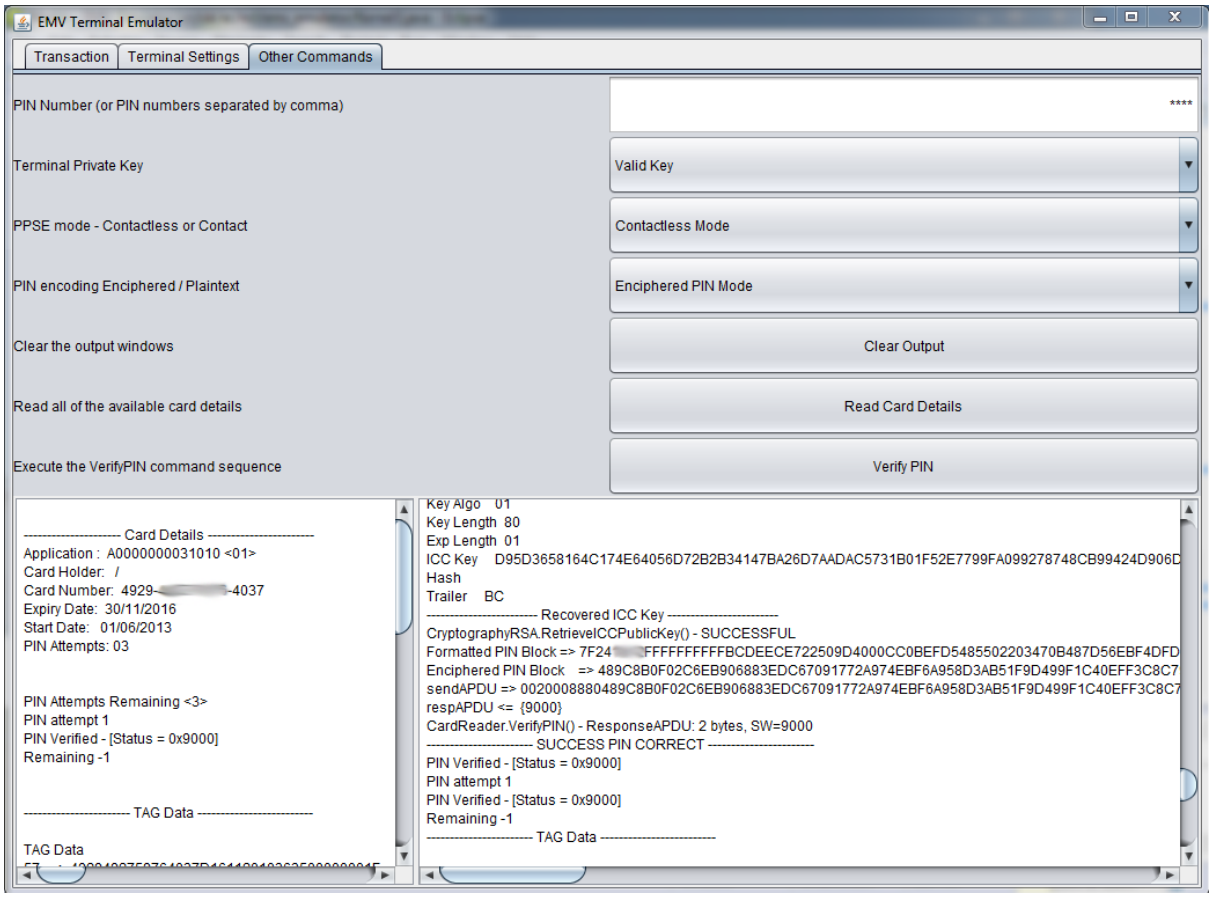


Figure 24 - POS Emulator Running the Contactless PIN Verify Protocol Sequence

Figure 24 shows the POS emulator running the PIN verify protocol sequence in contactless mode. The log trace (bottom left) shows that the card verifies the enciphered PIN which is an operation that it should not perform in contactless mode (see Chapter 7 for details of the contactless Verify PIN vulnerability).

5.5.3 Implementation of the Protocol Emulator

The protocol emulator is a Java Application for a PC platform. The emulator uses a PC/SC compliant USB smartcard reader to communicate with the both contactless and Chip & PIN EMV cards. The

software uses the javax.smartcardio library to control the PC/SC smartcard reader and communicate with the card.

The emulator code is designed to closely follow the structure of the EMV specifications. There is a Java class for each of the protocol sequence variants (kernels) described in the the EMV specification [6] [7]. There is an Entry Point class which decides which of the kernels will be performed based on the capabilities of the EMV card, this reflects the functionality described in [7] Book B Entry Point Specification. The structure of the protocol emulator includes the following:

- entry point processing
- contact "Chip & PIN" transaction protocol, which we have called kernel 0 in the emulator
- kernel 1 - Visa contactless transaction protocol with online transaction authorisation support
- kernel 2 - MasterCard contactless transaction protocol
- kernel 3 – Visa fDDA contactless transaction protocol for offline only transactions
- kernel 4 - American Express contactless transaction protocol,
- kernel 5 – JCB contactless transaction protocol
- kernel 6 - Discover contactless transaction protocol
- kernel 7 – UnionPay contactless transaction protocol.
- contactless magnetic stripe transaction protocol

This emulator uses an abstract kernel class containing the common functionality, with each of the kernels being a concrete implementation of the abstract kernel class and adding the card provider specific functionality.

5.5.4 Protocol Emulator Structured Coding

The protocol emulator uses structured comments in the code to provide a linkage between the running software in the emulator and the:

- the UML diagrams: `// UML 9.7 Transaction Approved Application Cryptogram (AC), AFL`
- the reference table, the emulator code contains the full descriptive text as per Figure 19:
  `// If the card approves the completion of the transaction in offline mode, it will...`
- the EMV Specifications: `// EMV v4.3 Book 3 - Annex B Rules for BER-TLV Data Objects - p155`
- the emulator's log traces: `Log.Write("UML 9.7 Transaction Approved", Log.PROTOCOL);`

In this way each line of Java code can be traced back to its origin in the EMV specification and can also be understood as part of the overall protocol sequence thanks to the references to the UML protocol sequence diagram.

```
Log.Write("UML 9.0 GetProcessingOptions(PDOL) ", Log.PROTOCOL);
// UML 9.0 GetProcessingOptions(PDOL data)
// In the Visa fDDA transaction Get Processing Options (GPO) is used to request completion
// of the transaction.  The PDOL data must contain all of the data elements requested by
// the card otherwise the transaction will be rejected
// EMV v2.2 Book C-3 - 2.4.1 Initiate Application Processing - p12
// EMV v2.2 Book C-3 - 5.2 Initiate Application Processing - p40
// EMV v2.2 Book C-3 - 5.2.1 Get Processing Options (GPO) Command - p40
// EMV v2.2 Book C-3 - 5.2.2 Initiate Application Processing - p40 to p46
// EMV v4.3 Book 3 - 6.5.8.4 Data Field Returned in the Response - p60
ResponseAPDU response = this.Reader.GetProcessingOptions(dol);
if (response.getSW() == Const.SW_SUCCESS)
{
   // Split the HEX string response from the card into individual fields with a TAG and Value
   // EMV v4.3 Book 3 - Annex B Rules for BER-TLV Data Objects p155
   // Or if it isn't TLV decode it as a Format 1 object –
   // EMV v4.3 Book 3 - 6.5.8.4 Data Field Returned in the Response Message
   if(!this.CardData.DecodeResponse(response))
      this.CardData.FormatGPOResponse(response.getData());
   byte [] iad = this.CardData.FindData(Const.TAG_IAD);
   Log.Write("UML 9.7 Transaction Approved", Log.PROTOCOL);
   // UML 9.7 Transaction Approved Application Cryptogram (AC), AFL
   // If the card approves the completion of the transaction in offline mode, it will
   // return Transaction Cryptogram  (TC) in the Cryptogram Information Data (CID).
   // The card also returns all of the data elements required by the terminal to
   //  complete the transaction:
   // Signed Dynamic Application Data (SDAD) used by the terminal to verify that the
   // card has approved the same transaction that the terminal sent.
   // Application Cryptogram (AC) used in the completion of the transaction with the
   // Bank to validate that a valid card completed the transaction.
   // Application File Locator (AFL) contains the location in the card's file
   // structure where the terminal can read the data elements required to complete
   // the transaction.
   // EMV v2.2 Book C-3 - 5.4.3 Determine the Card Disposition - p50
   // EMV v2.2 Book C-3 - 5.2.2.2 GPO Response SW1 SW2 - p43
   // EMV v2.2 Book C-3 - 5.2.2.3 Contactless Path Determination - p46
   // EMV v2.2 Book C-3 - A.2 Data Elements by Name - p97
   // EMVv2.2  Book C-3 - Annex C Fast Dynamic Data Authentication - p127
   // EMV v2.2 Book C-3 - C.1 Dynamic Signature Verification - p128
   // EMV v4.3 Book 3 - 6.5.8 Get Processing Options APDUs - p59
   // EMV v4.3 Book 3 - 6.5.8.4 Data Field Returned in the Response - p60
   // TC Returned by Get Processing Options (9F10)
   if (Util.BitCompare(iad[4], Const.IAD_VISA_STATUS_MASK, Const.IAD_VISA_TC))
   {
      Log.Write("--------- TC Returned by Get Processing Options (9F10) -------", Log.PROTOCOL);
```

Figure 25 - Section of Protocol Emulator Code

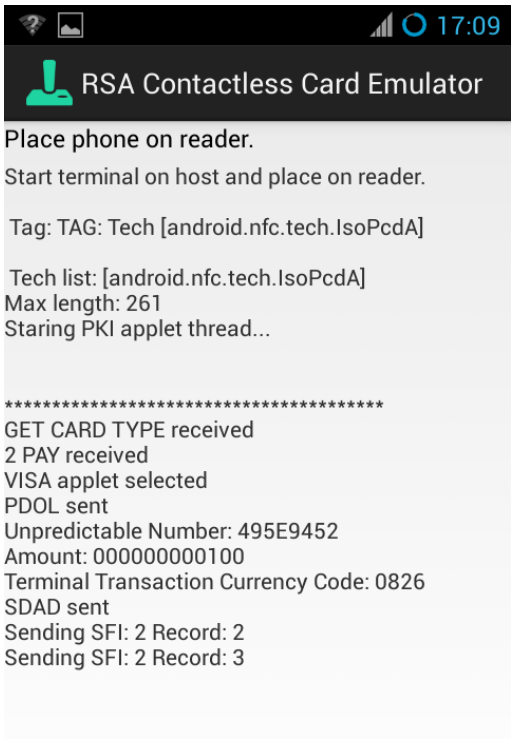
Figure 25 shows a section of POS emulator code which performs the operations which correspond to the UML diagram steps *9.0 GetProcessingOptions(PDOL data)* and *9.7 Transaction Approved - TC, SDAD, Application Cryptogram (AC), AFL*. The section illustrates the use of structured code comments and the use of log traces to record UML references into the record of the protocol message sequences (see Figure 22 for an example of the log trace output).

The protocol emulator is a concrete software implementation of the EMV transaction protocol. It is both an end product of the analysis methodology and also the test-bed used to validate the findings of our analysis methodology; for instance the protocol emulator was used to confirm the existence of the foreign currency flaw in UK issued payment cards.

5.5.5 EMV Card Emulator

The card emulator completes our ability to emulate of the both sides of the EMV contactless protocol sequence. The card emulator can be used in two ways:

- to implement working solutions for vulnerabilities in the EMV protocol sequences. The POS emulator and card emulator can be used together to demonstrate the efficacy of a proposed change to the contactless protocol e.g. the POS authentication protocol Chapter 8
- the card emulator can be used as surrogate card when developing new protocol sequence in the POS terminal emulator. This is a more efficient process than using real EMV cards which may get blocked by an incorrectly coded protocol sequence.



UML 4.0 ListAvailableNFCApplications()
"2 PAY received"
UML 4.1 List of available applications
the card emulator returns a list of payment applications, in this case "Visa"
UML 7.0 SelectApplication(AID)
"VISA applet selected" the card activates the Visa application and Visa data including the PDOL
UML 7.1 Transaction Setup data inc. PDOL
"PDOL Sent" the card returns the PDOL
UML 9.0 GetProcessingOptions(PDOL)
"Amount 000000000100"
"Terminal Currency Code: 0826"
UML 9.1 GenerateUPN()
"Unpredictable Number:096392C7"
UML 9.7 Transaction Approved TC + SDAD
"SDAD sent" if the card emulation approves the transaction it returns an SDAD to the POS
UML 10.0 ReadAFLRecord(SFI, record)
"Sending SFI 2 Record 2"
"Sending SFI 2 Record 3" the card emulator returns the data blocks requested by the POS

Figure 26 - Card Emulator Kernel 3 fDDA Protocol Sequence

Figure 26 shows the card emulator responding to the messages sent by a POS terminal during a Kernel 3 fDDA transaction protocol sequence. The left side of Figure 26 shows the screen output of the card emulator. The right hand side of Figure 26 shows the UML diagram references which correspond to the text displayed by the card emulator.

5.5.6 Implementation of Card Emulator

The card emulator was implemented by Joseph Hannon. It was written for an NFC enabled Android phone. This takes advantages of the Android mobile phone ability to operate in NFC card emulation mode [56] which allows the Android mobile phone to act like an EMV contactless payment card when it is presented to a POS terminal.

The card emulator currently implements kernel 3 (Visa) and kernel 2 (MasterCard) contactless card types. The card types share a great deal of functionality however, there are a number of rules and data structures which are particular to specific card types. For instance MasterCard has a state machine, Figure 37, which dictates the order in which commands can be executed in the kernel 2 protocol, this is not present in kernel 3 specification. Kernel 2 cards have a publicly accessible data structure which contains a record of the last 10 transactions whereas Visa implements a cumulative transaction total which is held privately by the card.

The card emulator is a passive listening daemon which waits to receive the protocol messages sent by the POS terminal and responds with an appropriate message and/or status code. It handles the following EMV protocol messages: SelectApplication(), GetProcessingOptions(), GetChallenge(), GetData(), ReadRecord(), VerifyPIN(), GenerateAC(), InternalAuthenticate(). The card emulator does not allow the methods to run concurrently which replicates the operation of EMV cards. It also enforces EMV card processing rules such as the SelectApplication() command which must be successfully completed before any other command can be executed. ExternalAuthenticate() is not currently supported by the emulator as it is not used in the kernels currently implemented in in the protocol emulator.

The card emulator includes the RSA and 3-DES cryptographic functionality to verify an enciphered PIN, generate an Application Cryptogram and generate a SDAD. To achieve this we have implemented a set of shared keys for the POS terminal emulator and the card emulator which follows the structure of the EMV PKI as described in section 3.3.4. This allows the card emulator to generate Application Cryptograms and SDAD signatures which can be validated by the POS terminal. The keys are self-signed so will only work between the POS emulator and the card emulator.

5.6 Z Abstract Model

The Z abstract model was developed in collaboration with Dr Leo Freitas, based upon the model of the EMV protocol sequences I had developed in building the protocol emulator. This was a collaborative effort where I provided detailed knowledge of the EMV protocol specification which Dr Freitas codified into the Z abstract model. Dr Freitas also drew upon his previous work on the formal analysis of Mondex smartcard payments system [57] to create the abstract model.

The abstract model identifies potential vulnerabilities in the EMV protocol and develops test cases to be run on the protocol emulator based on the vulnerability. It was in the building of the pre-conditions of the Z abstract model that the foreign currency flaw was spotted (see section 5.8).

The description implementation of the Z Abstract Model in this section was taken verbatim from a paper co-authored with Dr Freitas [29]. I have left these two sections as written by Dr Leo Freitas because he is the creator of the model and his words best describe the implementation process.

5.6.1 Motivation for the Z Abstract Model

In this work, we studied the EMV requirements documents [6] [7] to produce a formal Z abstract model of its properties and functionalities, specifically for the kernel 3 fDDA contactless transaction protocol, summarised in Figure 18. The motivation is to capture these requirements mathematically, enabling us to check that the properties of interest hold (i.e. the requirements documents are consistent), and to produce test cases for our EMV emulator derived from formal proof of operational feasibility of each protocol stage (i.e. by proving the stage is feasible, we expose both abstract behaviours: normal and exceptional).

Our abstract model uses the Z notation [58], which Dr Leo Freitas had previously used to successfully model the Mondex card payment protocol [57]. Mondex was awarded ITSEC level E6 (ITSEC's highest security-level) [59], this can be directly attributed to the formal analysis of the Mondex protocol performed using Z notation and the Eves verification tool. The Mondex card payment protocol was developed by National Westminster Bank in the early 1990s, and later bought MasterCard where it was influential in the development of the EMV "Chip & PIN" protocol. Thus, the consideration to use Z for mechanising EMV was straightforward: we already had some aspects of the mechanisation (e.g. necessary proof engineering, organisation of tokens/certificates, auxiliary lemmas and definitions, etc) in place. Having said that, other formal languages with theorem proving support, such as Event-B or KIV would be equally suitable.

In 2006, 10 years after the release of Mondex, Prof. Woodcock led a Grand Challenge effort to identify any residual errors in the Mondex proof using automated theorem provers. In the process, nine different teams using different formalisms attempted the problem, where the work of Dr. Leo Freitas [60] was the "control" in the sense that it mechanised the original "warts-and-all" rather than with any translation to different languages [59].

5.6.2 Implementation of the Z Abstract Model

Proof obligations in Z are usually of three kinds: *well-formedness of models*, where partial functions are applied within their domains, and unique existential quantifiers are sound; *operational feasibility*, where specified operations have (implicitly defined) pre-conditions strong enough to establish (explicitly defined) post-conditions; and *data reification* via (usually forward) simulation, where the use of (concrete) data structure representations in operations closer to an implementation language are shown to respect the abstract representation and operations.

Our models have 49 type definitions, 61 Z schemas representing the NFC operations of the protocol, and 79 proofs in total, of which 49 are theorems representing properties of interest for the whole model [61]. Feasibility proofs are useful in deducing formal model-based test cases, as they characterise the complete space of behaviours for all operations of interest, including successful and all possible error cases, both determined by mathematical predicates representing disjoint behaviours

of the protocol. That is, feasibility proofs characterise a set of disjoint predicates with (in EMV's case) non-overlapping conditions that when accumulated lead to true (e.g. pre-condition of an operation being $x < 0$ or $x > 0$ or $x = 0$). Thus, each disjunct represents a unique class of behaviours for the functionality being proved. Moreover, we also prove that these disjunct predicates amount to true, hence we guarantee all behaviours are accounted for.

The formal model follows the methodology advocated in [62], which enumerates requirements realised by each piece for formal specification. Thus, if all elements of the requirements are accounted for within the abstract mathematical model in a way that conveys the intended behaviour described in English, then proofs about the abstract model (or rather, proof failure) will lead (as our experiments show) into potential attacks and vulnerabilities discovered through proof investigation. Once validated by EMV experts, such a formal model becomes a more accurate representation of the EMV protocol than the EMV books [6] [7].

These efforts correspond to the POS terminal side of Figure 18. The mechanisation of a formal concrete design, together with a proof of refinement indicate that these designs faithfully satisfy the abstract model linked to the requirements. Refinement proofs are perhaps the most costly aspect of a proof exercise, as it needs to establish that the implementation details do not breach any of the contractual requirements established by the abstract model. This concrete model can then serve to annotate the Java (or any other implementation) with formal specification for code-level functional correctness as done by tools such as VeriFast [63].

Furthermore, we derive a set of test cases from this abstract model that is the smallest with highest coverage possible. We also derive a systematic code-annotation technique, using the same principle to enumerate what aspect of the requirements each piece of code within the emulator is realised. These test cases represent a test-oracle based on requirements testing, rather than testing for any implementation issues. Together, the test cases and systematic code annotation are useful for capturing potential (major) errors. Errors from the concrete design are more likely to expose problems with implementation choices, and it is our aim in the future to annotate the emulator code with formal specification amenable to static analysis of the properties corresponding to the behaviour of the code.

5.6.3 Example: Section of Z Abstract Model

Figure 27 shows a section corresponding to *9.7 Transaction Approved - TC, SDAD, Application Cryptogram (AC), AFL* in the UML diagram. The section of Z specification describes the relationship between the SDAD and the PDOL. The PDOL is the transaction data sent to the card in the message represented by UML *9.0 GetProcessingOptionsCommand(PDOL data)*. The SDAD is part of the data returned by the card in the message depicted in the UML diagram *9.7 Transaction Approved - TC, SDAD, Application Cryptogram (AC), AFL.*

FS.SDAD.Algorithms

cardAuthRelData [9, p.90] *encrypt [9, p.134]*
hashCode [9, p.132], [2, p.68]

To create an *SDAD*, the card needs a few algorithms to be available. They are abstracted away here as functions from the needed input, coming from a *PDOL* or the card state, to the actual values needed for the terminal. Function *cardAuthRelData* generates a unpredictable number given a card id *CID*. Similarly, the *hashcode* function produces a hash value for the inputs: card unpredictable number, the *PDOL*'s unpredictable number, the *PDOL* amount, and the *PDOL* currency. Finally, all that gets encrypted using the card's transaction counter paired with this hashcode and the card's private key.

$$
\begin{array}{l}
cardAuthRelData : CID \rightarrow \mathbb{N} \\
hashCode : \mathbb{N} \times (\mathbb{N} \times (\mathbb{N} \times CURRENCY)) \rightarrow \mathbb{N} \\
encrypt : \mathbb{N} \times (\mathbb{N} \times SDAD) \rightarrow \mathbb{N} \\
decrypt : \mathbb{N} \times (\mathbb{N} \times \mathbb{N}) \rightarrow SDAD \\
\hline
\forall\, trCnt, puKey, prKey, issuerKey, cypher : \mathbb{N} \bullet \\
cypher = encrypt\,(trCnt, (prKey, decrypt\,(issuerKey, (puKey, cypher)))) \\
\forall\, trCnt, puKey, prKey, issuerKey : \mathbb{N};\ msg : SDAD \bullet \\
msg = decrypt(issuerKey, (puKey, encrypt\,(trCnt, (prKey, msg))))
\end{array}
$$

theorem rule lEncryptResType

$$\forall\, trCnt, prKey : \mathbb{N};\ msg : SDAD \bullet encrypt\,(trCnt, (prKey, msg)) \geq 0$$

theorem rule lCardAuthDataResType

$$\forall\, cid : CID \bullet cardAuthRelData\,(cid) \geq 0$$

theorem rule lHashCodeResType

$$
\begin{array}{l}
\forall\, cUpno, pdolUpno, pdolAmnt : \mathbb{N};\ pdolCurrency : CURRENCY \bullet \\
\quad hashCode\,(cUpno, (pdolUpno, (pdolAmnt, pdolCurrency))) \geq 0
\end{array}
$$

We create an *FDDA NFC SDAD* output, providing apprpropriate *PDOL* with corresponding message payload. We also need to check the card information is correct with respect to the card's requested *PDOL* data responded by the terminal.

FDDANFCVisaSDAD

$$
\begin{array}{l}
NFCVisaPDOL?;\ Card?;\ NFCVisaSDAD! \\
\hline
cstart? \leq_D pdolDate? \leq_D cend? \\
pdolAmount? \leq climit? \\
pdolAmount? \leq nfcOfflineCounter? \\
0 < contactCounter? \\
\\
sdadCardTrCounter! = transacCounter? \\
sdadHashCode! = hashCode\,(cardAuthRelData\,(cid?), \\
\qquad (pdolUpno?, (pdolAmount?, pdolCurrency?)))
\end{array}
$$

These include the card start/end date with respect to the transaction date, the card limit and the transaction amount, the card conctactless (*NFC*) counter threshold (*i.e.* when it reaches zero, we need to go online). For the *SDAD*, we need to encode the current transaction counter from the card, as well as a hashcode with information relating the card *CID*-based unpredictable number, and the corresponding unpredictable number, amount and currency from the terminal's *PDOL* response.

Figure 27 - Example Section of Z Abstract Model

In the Z Abstract model the referencing notation is slightly different to that seen in the Protocol Emulator code and reference table e.g. *[9. p.134]* where *9* is a document reference to EMV Book C-3 version 2.1 and *p.134* is the page number.

The full Z abstract model can be read in [61] which contains the Z specification and the proofs.

5.7 The Process of the Analysis Methodology

To address the complexity of the EMV specifications we have developed a systematic analysis methodology which combines formal and informal techniques. I have described the individual

elements which comprise the methodology, UML diagrams, reference tables, protocol emulator and Z abstract model. In this section I will describe how those elements are used together to create our analysis methodology.

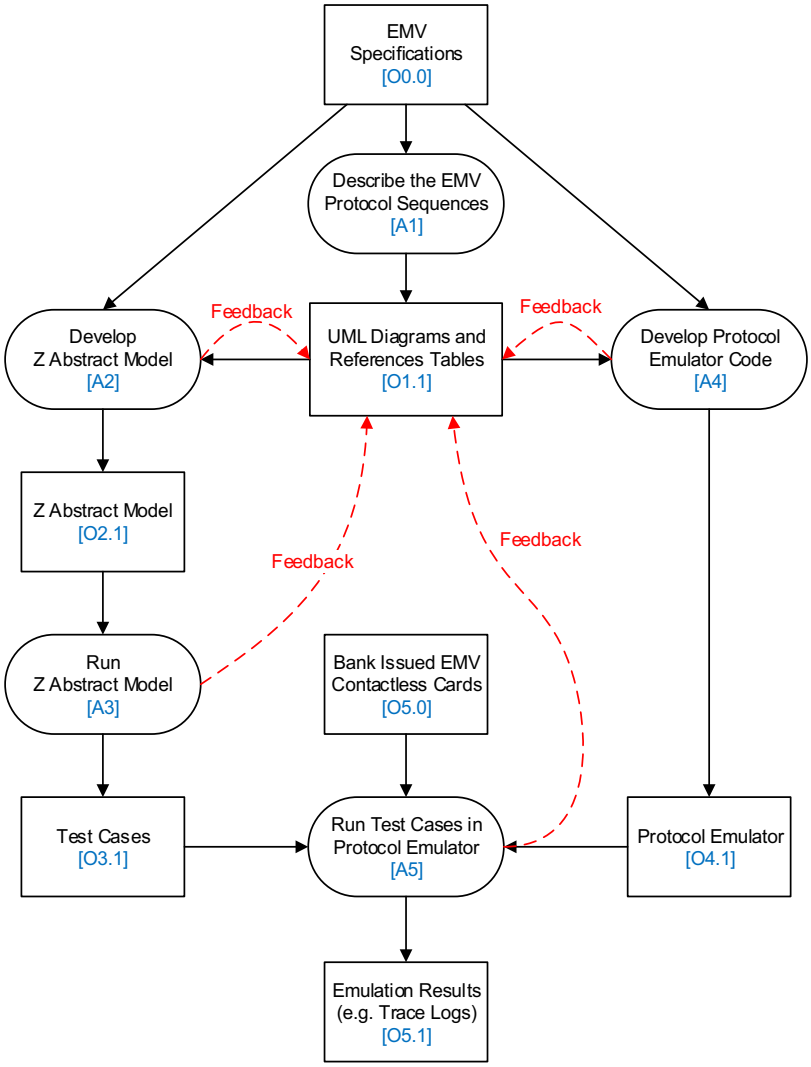


Figure 28 – Process Flow of Analysis Methodology

Figure 28 shows process flow which connects the separate elements (UML diagrams, reference tables, protocol emulator and abstract model) of our analysis methodology. The rounded boxes are activity nodes within the process e.g. [A1]. The square boxes are object nodes e.g. [O1.0]: these are the data sources that drive the activities. Connecting edges, represented as black solid-arrows, indicate the default order in the flow of activities. The red dashed-arrows are connecting edges that indicate feedback, creating an iterative process of refinement of the UML diagrams [O1.1], the Z abstract model [O2.1] and the protocol emulator [O4.1].

The EMV specifications [O0.0] are the originating source of all of the data in the process. Any data or assumption made in the emulator code or in the abstract model should be traceable back to its origin (i.e. the book/section/page within the EMV specifications). The EMV specifications are structured so that the complete description for a single transaction protocol sequence is split across

multiple sections and multiple books. The UML sequence diagrams [O1.1] collate the multiple books of the EMV specification, into a single easy to follow description of the transaction sequence. The transaction sequence diagrams [O1.1] are the initial stage of the iterative process that we used to create the concrete software implementation of the emulator [O4.1].

Much of the process is concerned with constructing the UML sequence diagrams as accurately as possible. To achieve this, we use a detailed analysis of the EMV requirements and a detailed working knowledge of the structure of the various specifications contributing to a single transaction. Moreover, we use feedback from the Z abstract model construction [A2], the derivation of test cases [A3] and the development of the protocol emulator [A4].

At each stage of the process if additional information is found about the working of EMV it is fed back into the UML transaction sequence diagrams [O1.1]. The feedback is essential to refine our understanding of the EMV specifications and to document it. Each time the UML diagrams are updated this drives the improvement of the coding of the protocol emulator [O4.1]. The protocol emulator is used in practical experiments [A5], running full or partial transaction protocol sequences against real bank cards [05.0] and generating results in the form of detailed log traces [05.1] which contain references to the UML diagrams for traceability.

5.8 Example of Using the Methodology

This section briefly describes the use of the analysis methodology in the identification and confirmation of the foreign currency flaw in the EMV contactless protocol. Identification of the flaw came from the Z abstract model and confirmation that the flaw was exploitable was provided by the protocol emulator.

The first indication that there was a potential issue came during the process of defining the pre-conditions of the Kernel 3 fDDA protocol sequence. Our method of working involves Dr Freitas building the Z specification based on my knowledge of the EMV specifications and Dr Freitas's knowledge of modelling.

Our first revision of the model highlights that there were a number of unsatisfied pre-conditions one of which was the relationship between the transaction currency and the local currency of the card. To satisfy this missing pre-condition I went back to the EMV specifications and located all references to the transaction currency and card currency. This yielded a single reference in EMV specification, Book 3 (version 4.3) [6] page 163, which states *"If transaction is in the application currency and is under X value"*, where X is the card offline transaction limit.

It was now clear from the abstract model that the currency pre-conditions could not be met when the transaction currency did not match the card's local currency. Figure 29 shows the section of the

model which deals with the relationship between the transaction currency *tcurrency* and the card's local currency *cardCurrency*.

$$
\begin{array}{|l}
\hline
FDDANFCVisaPDOL \\
NFCVisaPDOL!;\ Transaction!;\ cardCurrency? : CURRENCY \\
\hline
pdolAmount! = convertCU\ ((tcurrency!, cardCurrency?), amount!) \\
pdolCashback! = cbamount! \\
pdolUpno! = tunpredictableNumber! \\
pdolCountry! = tcountry! \\
pdolCurrency! = cardCurrency? \\
pdolDate! = tdate! \\
pdolTrType! = type! \\
\hline
\end{array}
$$

Figure 29 - Abstract Model Z Schema

Once the abstract model had identified the potential flaw the protocol emulator was used to verify the existence of the flaw and quantify the degree to which the flaw is exploitable.

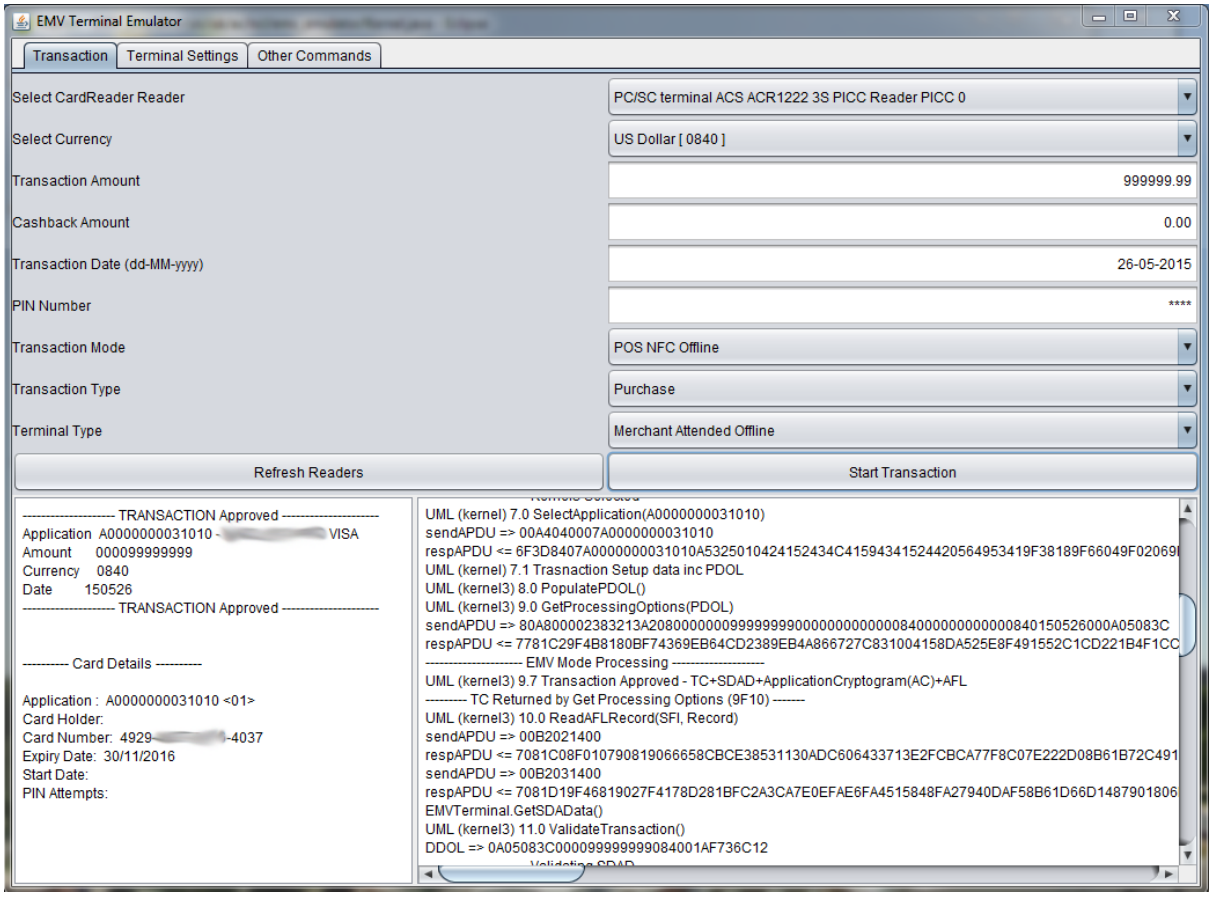


Figure 30 – POS Emulator Running the Foreign Currency Flaw Protocol Sequence

A protocol sequence was implemented in the POS terminal emulator that fully exercised the flaw. From this I discovered that the foreign currency flaw was present and exploitable in most of the Visa cards tested. I found that the amount authorised by Visa debit cards, varied from $0.00 to $999,999.99, depending upon the Issuing Bank and the type of card (debit or credit).

Figure 30 shows the POS terminal emulator running the foreign currency protocol sequence. The result of this test is that the contactless Visa card authorised the amount $999,999.99, thereby showing that the vulnerability was present.

The model indicated that the vulnerability was exploitable in kernel 3 (Visa) fDDA contactless protocol due to kernel 3 having the offline only transaction authorisation. Kernels with online transaction authorisation such as kernel 2 (MasterCard) were not affected by the vulnerability as the card would request online completion of the transaction if a foreign currency was requested. To verify this, I tested several kernel 2 contactless cards using the foreign currency protocol sequence and found in all cases that the kernel 2 cards rejected the protocol. This can be attributed to the difference in authentication processes of the two protocols, section 3.6.

5.9 Conclusion

Our analysis methodology combines the strengths of formal analysis with a functional model of the protocol. The process allows us to:

- identification of vulnerabilities in the EMV protocol specifications.
- perform exhaustive experiments on the EMV protocol which clearly define the scope and impact of the vulnerability.
- provide a documented link between the vulnerability observed in real EMV cards and the origin of the vulnerability in the EMV specification.

Development of the protocol emulator and the Z abstract model have been carried out in parallel, with feedback from the development of one influencing the development of the other. This has resulted in a deeper understanding of the EMV protocol sequences and thereby more exact representation of the specification by the protocol emulator. A practical example of this process follows; in the Z abstract model the card CVM and terminal capabilities are modelled as sets of capabilities, for which one of the possible options is an “empty set”. This makes a practical difference in the protocol emulator where CVM of “0000” is very different from the empty set of “ ”.

For the most part, knowledge gained in the development of the UML diagrams and the protocol emulator was used to develop the Z abstract model. However, there were many cases where the development of the abstract model highlighted gaps in the knowledge captured in the protocol emulator, deriving from the abstract model’s ability to highlight the absence of value.

The methodology described in this chapter has been the foundation on which our contactless payments research has been based. It creates a structured framework that can identify vulnerabilities, demonstrate that they are exploitable in the real-world and link the vulnerability back to a specific point in the original EMV specification. Which is distinct from the identification of implementation error in particular cards or POS terminals

Chapter 6. Foreign Currency Flaw

This chapter describes the "Contactless Foreign Currency Flaw" which was identified in the EMV contactless protocol specifications using our analysis methodology, Chapter 5. The flaw in the specification translates into an exploitable vulnerability in UK issued contactless Visa cards, which allows the attacker to bypass the safeguards built into the EMV contactless protocol.

The chapter is organised as follows; in 6.1 we outline the vulnerability and highlight the EMV security features bypassed by the attack, in 6.2 we outline the safeguards built into the EMV payments system and in 6.3 explain the exploited EMV functionality. in 6.4 we briefly describe how the vulnerability was discovered, in 6.5 we outline the attack scenario which exploits the vulnerability, in 6.6 we describe the experimental implementation carried out to demonstrate the real-world impact of the vulnerability, in 0 we give the test results of our experimental work and in 6.8 and 6.9 we explain our conclusions and propose a potential solution.

6.1 Vulnerabilities Arising from Foreign Currency Flaw

Our research has identified a practical attack on EMV contactless credit and debit cards, which allows large-scale "harvesting" of fraudulent payments from unsuspecting cardholders. The attack exploits six functional characteristics of EMV contactless credit and debit cards:

- UK issued kernel 3 (Visa) credit cards *will approve transactions values of 999999.99 in any foreign currency*; this allows the attack ignore the £20 contactless payment limit.
- The *contactless interface* allows transactions to be extracted whilst the card is still in the cardholder's wallet.
- The cardholder's *PIN is not required for contactless transactions*; this allows the fraudulent transaction to be extracted from the card without any further interaction from the cardholder.
- Kernel 3 contactless cards will approve *transactions in offline mode*; this allows the attack to be performed without connecting to the card payment system, thereby avoiding any additional security checks by the bank.
- The *merchant details* are *not included* in the data cryptographically protected by the card; this allows the merchant details to be added later, making the attack more flexible and scalable.
- While the EMV protocol requires payment cards to authenticate themselves to the POS terminals, currently *there is no requirement for POS terminals to authenticate themselves.*

The main contribution of this chapter is the identification of a newly discovered vulnerability of the EMV protocol centred on the card's handling of foreign currencies. This is made possible by a combination of the six functional characteristics described above. The introduction of EMV contactless cards has created a situation comparable to that described by Reason in his "Swiss cheese" model [64] where layers of protection can be compromised if holes on each layer line up to create an exploitable attack. In this case, the six characteristics line up in a way that defeats the safeguards put in place by EMV. Through this chapter we also contribute two potential solutions which will mitigate this vulnerability.

The ability to capture fraudulent transactions and store them for later transmission to a rogue merchant makes this attack different from previously described relay attacks [42] [46] [44] on EMV contactless cards. The relay attack depends upon very close synchronisation between two attackers; the first attacker has to be in contact with the victim's card whilst the second attacker makes a purchase at a POS terminal. This makes relay attacks difficult to operate on a large scale.

Similar to Murdoch et al. (2010) [13], our attack can potentially be operated on a large scale. Murdoch et al. (2010) allows attackers to buy goods from retailers, whereas the attack described in this chapter is different in that it targets the money in the victim's bank account.

The very recent Bond et al (2014) [25] attack is similar to our attack in that it could be operated on a large scale and it extracts money from the victim's account. It would be interesting to explore the possibility of using our mobile phone contactless-transaction-collecting app as the "skimming" platform for the attack.

6.2 EMV Transaction Safeguards

In the UK, EMV credit / debit cards can perform two different transaction types: contactless transactions, and contact (Chip & PIN) transactions.

6.2.1 Contactless Safeguards

Contactless transactions are intended to be a fast and convenient replacement for small cash purchases. In a contactless payment, the credit / debit card is placed on the POS terminal's contactless reader for approximately 1 second and the payment is approved.

There are two significant differences between a contactless transaction and a contact Chip & PIN transaction. First, the contact transaction requires the cardholder to enter their PIN, whereas the PIN is not required for contactless transactions. Second, contact transactions require the card to be removed from the wallet and inserted into the POS terminal, whilst a contactless transactions is completed wirelessly by placing the card on the POS terminal, this can be done whilst the card is still in the wallet.

PIN entry provides one of the key safeguards in Chip & PIN transactions. The PIN ensures that only the cardholder, who knows the PIN, can use the card. Contactless transactions are not protected by PIN entry. EMV have therefore implemented the following safeguards to limit the potential loss from lost or stolen contactless cards:

In the UK, each contactless transaction is limited to £20; any transaction above this value will require a Chip & PIN transaction.

EMV cards are limited to five consecutive contactless transactions, after which the PIN must be entered in a Chip & PIN transaction.

These safeguards ensure that the maximum loss due to a lost or stolen contactless card is £100.

6.2.2 Contact Chip & PIN Safeguards

The majority of EMV card transactions are Chip & PIN transactions. Chip & PIN transactions allow purchases up to the balance of a debit card or the credit limit of a credit card.

Chip & PIN transactions are protected by the following safeguards. First, the cardholder must enter their PIN to authorise the transaction. This is used to ensure that the person making the payment is the authorised cardholder.

Second, if the value of the transaction is greater than the card's *offline* transaction limit, the card will request that the POS terminal makes an *online* connection to the bank to perform additional authorisation checks. The POS terminal must connect to the bank to provide the card with the *online* authorisation code (Authorisation Response Cryptogram (ARPC)). The bank will respond with the authorisation code only if the card has not been reported lost or stolen, and the account has sufficient funds to pay for the transaction. The card will only authorise the transaction if it receives a valid online authorisation code from the POS terminal.

6.2.3 Cryptographic Protection of Transactions

The EMV payment system utilises cryptography to ensure that (i) only genuine EMV credit / debit cards can authorise transactions (ii) the transaction details approved by the card cannot be altered.

Application Cryptogram (AC)

The AC contains a Message Authentication Code (MAC). The MAC utilises a symmetric algorithm, either 3-DES or AES, to encipher the transaction data fields detailed below:

- amount authorised (value of the purchase)
- amount other (cashback amount if required)
- terminal country code (UK - 0826, USA - 0840 etc.)
- terminal verification results (POS status code)
- transaction currency code (UK£ - 0826, US$ - 0840 etc.)

- transaction date
- transaction type (purchase - 00, cash - 01, refund - 20)
- POS terminal unpredictable number (prevents cloned cards)
- application interchange profile (card's security capabilities)
- application transaction counter (card's Application Transaction Counter (ATC))

The AC is sent to the bank as part of the Financial Presentment message, Table 7. This allows the bank to verify that the transaction details supplied by the merchant are the same as the transaction approved by the EMV card, section 3.5.

Signed Dynamic Authentication Data (SDAD)

The SDAD is a RSA digital signature on a SHA1 hash of the transaction data. In the Kernel 3 fDDA protocol the transaction data included in the SDAD are:

- POS terminal unpredictable number
- amount authorised
- transaction currency code
- card unpredictable number
- card transaction qualifiers

The SDAD is used by the POS terminal to verify that the card is genuine, section 3.3.2.

6.3 EMV Functionality Exploited by the Attack

The attack circumvents the safeguards built into EMV credit / debit cards by exploiting some EMV functionality that has been made vulnerable due to the introduction of contactless payment interface. In particular, there are three features that are exploited in our attack scenario:

- Contactless foreign currency transactions. As described in section 6.2.1 the safeguards built into EMV will limit the maximum value allowed for each contactless transaction to £20. Any amount over £20 will require the cardholder to enter their PIN, and any amount above the *offline* transaction limit (e.g. £100) will require the POS terminal to connect to the bank to perform additional checks before the transaction is approved. Our research has found that EMV credit and debit cards can be tricked into approving contactless transactions of much higher value than £20, simply by requesting the transaction in a foreign currency. In our experiments, EMV cards have been found to approve contactless transactions up to €999,999.99 without requesting the PIN, and without requesting that the POS terminal goes online to perform additional checks. This sidesteps the usual safeguards employed by EMV payments system.

- Wireless interaction with card. This attack exploits the wireless interface on contactless cards to collect transaction authorisations whilst the card remains in cardholder's wallet. This means the cardholder remains unaware that they have been exploited until their card statement arrives, thereby allowing the attack to operate for longer and be more lucrative to the attackers.
- The merchant ID and terminal ID can be added later by the rogue merchant, as these data are not included in the AC generated by the card. The AC cryptographically ensures that the transaction data approved by the card, Table 6, is the same as that received by the Issuer.

6.4 Identification of the Foreign Currency Flaw

The identification of the foreign currency flaw in the kernel 3 (fDDA) contactless transaction protocol demonstrate the efficacy of our analysis methodology, Chapter 5. The discovery of this particular vulnerability can be attributed to the process of establishing the pre-conditions for Z abstract model.

In layman's terms, the abstract model tells us that the outcome of a Kernel 3 fDDA contactless transaction is dependent on the values of certain fields at the start of the transaction i.e. the *"pre-conditions"*. In this case the pre-conditions of interest are:

(α) Value of the transaction

(β) Card offline transaction limit

(γ) Currency of the transaction

(δ) Native currency of the card

The abstract model is a set of rules which are defined and based on the EMV specification. In this case the rule is based on the EMV specification, Book 3 (version 4.3) [6] page 163, which states *"If transaction is in the application currency and is under X value"*, where X is the card offline transaction limit.

This creates a rule in the abstract model $(\alpha) < (\beta)$ which can only be evaluated if (γ) is equal (δ). However, the EMV specification does not state what should happen when (γ) not equal (δ) (i.e. the transaction is in a currency foreign to the card). The abstract model reports this as a gap in the rule-set; identifying that there is a given set of pre-conditions for which there is no corresponding rule in the abstract model. The gap in the abstract model thereby identifying a potentially exploitable error in the EMV specifications.

6.4.1 Software Emulation of the Flaw

Having identified a potential flaw in the specification our software emulator allowed us to run the different test case scenarios exploring what would happen for the different card types.

It was found that kernel 2 cards would request online authorisation of the transaction, using the ARQC online request cryptogram, when the POS terminal requested an offline transaction in a foreign currency. This prevents the attack as the Issuing bank will know the correct exchange rate for the foreign currency and prevent the high value foreign currency transaction.

However, we found that in the kernel 3 (fDDA) contactless protocol sequence, the card simply returns the TC Application Cryptogram on the first request. Thereby missing out on the opportunity for the Issuing Bank to spot the high value foreign currency contactless transaction and block it.

We carried out testing on UK contactless kernel 3 (Visa) cards; finding that the exploitable vulnerability was present in the majority of UK issued cards.

Once we had used the emulator to find that the vulnerability was present in UK issued contactless cards, we built an attack scenario, section 6.5, to demonstrate that the vulnerability could be exploited in the real-world and was not just a problem restricted to the computer lab.

6.5 Attack Scenario

Figure 31 illustrates the attack scenario described in this section.

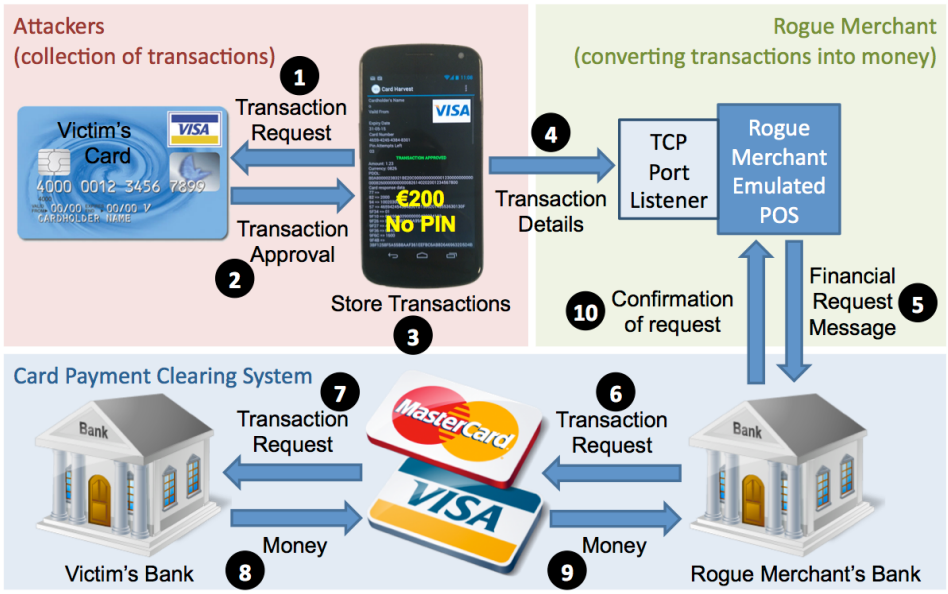


Figure 31 - Transaction Harvesting Attack

The attack consists of two stages:

Attackers (collection of fraudulent transactions): attackers using Near Field Communication (NFC) enabled Android mobile phones can collect fraudulent transactions from unsuspecting cardholders. This can be done whilst the contactless card is still in the cardholder's wallet (see steps 1 to 3 of Figure 31).

Rogue merchant (converting transactions into money): a rogue merchant converts the collected transactions into money in their bank account by sending the transaction data to a bank (steps 4 to 5 of Figure 31).

Finally the transaction request enters the *Card payment clearing system* where the rogue merchant's bank acts innocently to transfer the transactions into the card payment system, which transfers the money from the victim's bank account into the rogue merchant's bank account (see steps 6 to 10 of Figure 31).

6.5.1 Collecting Fraudulent Transactions

Transactions are collected using a malicious app written for NFC-enabled Android mobile phones. The app automatically initiates and collects a transaction immediately upon detection of a contactless credit / debit card in the phone's NFC field. This process takes less than 500 milliseconds from card detection to transaction completion.

It is imagined that attackers will operate in a similar way to pickpockets, hiding their activity in crowded situations such as on public transport or in the crowd at an event. When a credit / debit card is detected, the app gives the attacker an audible signal through their headphones; a second audible signal is given when the transaction collection is complete. This will allow the attacker to operate without attracting too much attention.

Hardware: An Android mobile phone is chosen as the attack platform for the following reasons:

- Android mobile phones have a built-in NFC reader.
- the Android phone is an innocuous item for the attacker to carry in a crowded place; for example, it will not raise attention if the attacker is stopped by the police, since everyone carries mobile phones these days.
- the Android mobile phone platform provides portability, Internet connectivity and good battery life, making it a very capable attack platform.
- access to the NFC functionality is open in the Android SDK making it easy to create Android apps which can access contactless EMV cards.

The Transaction Collecting App: The attack starts when the NFC-enabled Android phone identifies a contactless credit / debit card which is vulnerable to this attack in the victim's wallet. The app sends a transaction request to the vulnerable card.

The app plays an audible alert to the attacker to signal that a vulnerable card has been found.

When the victim's card receives the transaction request message, it can approve or decline the transaction. If the card approves the transaction it generates the AC and the SDAD, this proves to the bank and POS terminal respectively that the card that approved the transaction was genuine.

The cryptographic algorithms used to generate the AC and SDAD also ensure that the transaction details cannot be changed subsequent to the card authorising the transaction.

When the attack is complete the app plays a second audible alert.

Storage of Approved Transactions: The app was designed to operate in locations where an Internet connection is not always available, for example on underground public transport. Therefore, the app will initially store the transaction data returned and when an Internet connection is available, will send the stored transaction data to the rogue merchant who will convert the transaction data into money.

The ability to capture fraudulent transactions offline and store them for later transmission is one of the novel features of this attack. This allows the attack to be operated on a large scale without the need for synchronisation.

Furthermore, storing the transactions minimises the time required to collect fraudulent transactions as the app does not have to wait for a connection. It also allows the attackers to operate in victim-rich crowded places that are normally without an Internet connection such as on subway trains, on buses and at large events.

Converting Transaction Data into Money: The criminals would set up a rogue merchant account with an acquirer bank in one of the countries that accept EMV payments. This rogue merchant will receive the fraudulent transactions collected by the attackers and convert them into money by sending the transaction data to the bank.

The rogue merchant consists of three elements:

- An Internet-based listening service, which will receive collected transaction data from attackers.
- A data format conversion process, which converts the fraudulent transactions collected by the attackers into the format required by the bank.
- A rogue POS terminal, which must imitate the actions of a legitimate POS terminal so that it does not raise the bank's suspicion. To achieve this, the rogue POS takes the previously converted data, adds the merchant data and sends that data to the bank using an Internet Protocol (IP) connection.

Internet-Based Listening Service: The rogue merchant provides an Internet-based listening service on a pre-arranged IP address and port number, to receive the fraudulent transactions from the attackers. The transactions are initially stored to be processed later, once the merchant details have been added to the transaction and the connection to the acquirer bank is available.

Data Format Conversion Process: Financial presentment request messages are used to transmit EMV credit / debit card transactions between the merchant (who captured the transaction) and the acquirer bank (who will process the transaction).

Merchant-related data such as merchant ID, terminal ID and the merchant's bank account details are added to the transaction to complete the data required by the EMV card clearing system. The fraudulent transaction is now ready for transmission to the acquirer bank.

The exact format of the message will differ slightly between different acquirer banks. However, there are a number of mandatory fields that are the same for every acquirer bank. Standard 70 [65] in the UK and ISO 8583 [66] in other EMV countries define the mandatory data fields which must appear in the financial presentment request message and the optional fields which may differ between the acquirer banks.

The software for our attack prototype implements a Standard 70 message format, complete with all of the mandatory fields and a number of optional fields (6.6).

6.5.2 Rogue POS Terminal Process

Once correctly formatted, the financial presentment request message is sent to the bank. The acquirer bank returns a financial presentment response message, to which the merchant responds with a financial presentment confirmation message that acknowledges receipt of the acquirer's response message.

The supported communication options for this message exchange are PSTN, X25 over ISDN, IP over ISDN, and IP over public networks (i.e. the Internet) for transmission of messages between the merchant and the acquirer bank. The software implementation presented in this chapter uses IP over the Internet.

Our software implements data format conversion, section 6.6.3.2, and implements the sending of the financial presentment request message, Table 7 over an IP connection protected by SSL/TLS encryption.

For obvious reasons we were unable to check against a real bank. Of course, one approach to defeating the attack is to try to detect rogue POS behaviour at the bank, but it is not clear how well this can be done. A simple solution would be to have the payment card reject any contactless foreign currency transaction immediately, but is just not practical. A more effective solution can be implemented by either forcing foreign currency contactless transactions to be carried out in online mode only, or where that is not possible, to switch the transaction to Chip & PIN.

6.6 Implementation

To validate our research, we have implemented a number of software elements which demonstrate the viability and practicality of the attack. The software consists of three separate applications:

- An Android mobile phone app which captures transactions from the cards. Transactions are stored on the Android phone to be transmitted to the rogue merchant later.

- A rogue merchant Internet listening service which waits to receive the captured transactions from attackers using the Android mobile phone app.
- A rogue merchant bank communications module which packages the transactions into financial presentment request messages for transmission to the bank. This module handles all of the communication with the bank, which involves sending the financial presentment request messages and receiving acknowledgement messages.

6.6.1 Android Transaction Capture App

We have implemented the attack platform on an NFC enabled Android mobile phone as this would be an innocuous device for an attacker to carry around in a crowd.

For implementation and testing, we selected the Google Nexus 5 mobile phone. Implementing on a mobile phone platform limits the effective range to approximately 1 cm. However, in testing the Nexus 5 was capable of extracting transactions from an EMV contactless card which was located in a leather wallet in the pocket of a pair of jeans worn by our "unsuspecting" test victim.

6.6.2 Android App Operation

The attacker starts by pre-setting the amount and currency for all the transactions which will be captured from the victims cards (Figure 32).

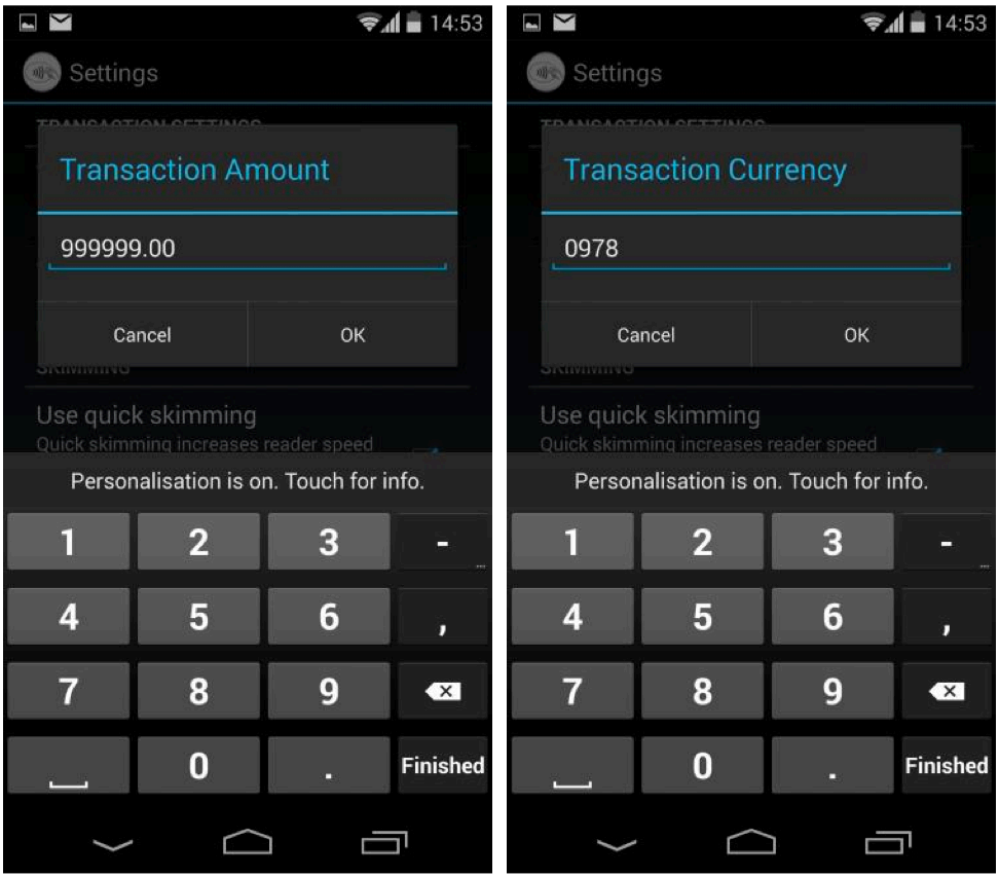


Figure 32 - Transaction Harvesting Settings

Figure 32 shows the attacker setting the amount to 999,999.00 and setting the currency code to 0978 which is the code for Euros. In testing we have also obtained transaction approvals in for 999,999.99 US Dollars, currency code 0840.

The app is now ready and will automatically collect a transaction from every EMV contactless card that it detects, without any further interaction from the attacker. This will minimise the chance of the attacker being detected, as they are not constantly interacting with their phone.

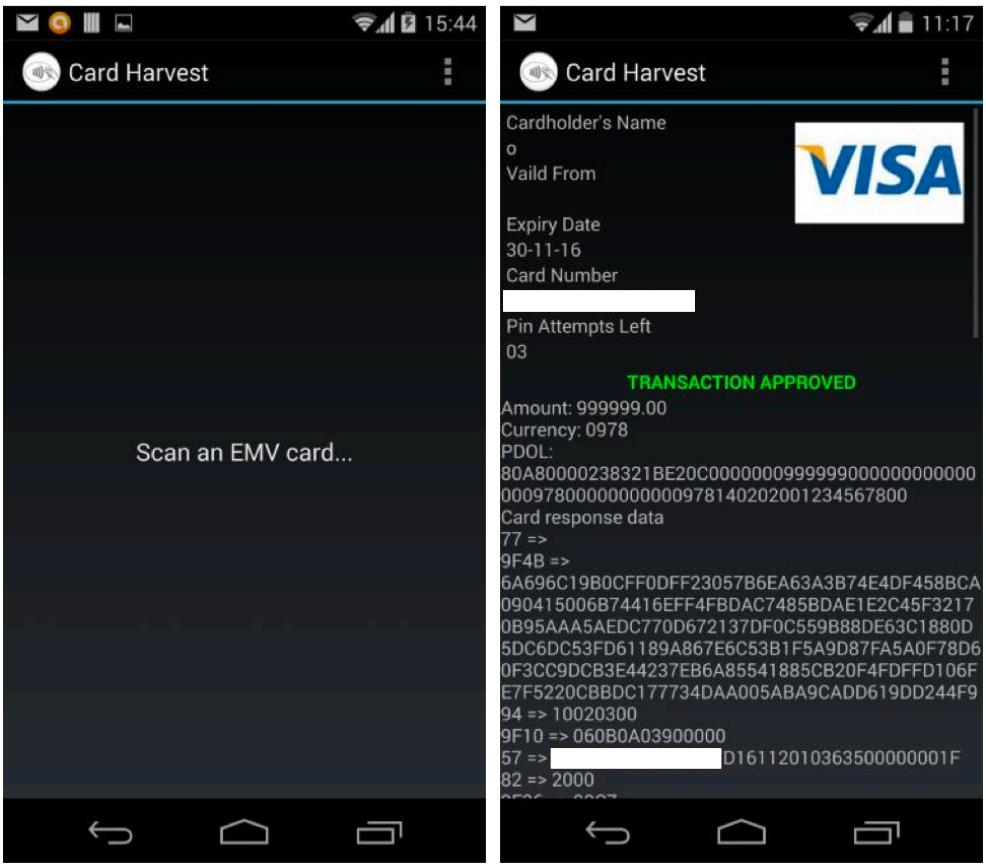


Figure 33 - Capturing the Transaction

In Figure 33 the screen on the left shows the app waiting to detect an EMV contactless card. The screen on the right shows the €999,999.99 transaction being captured from the card.

When the app detects an EMV contactless card, it sounds an audible alert in the attacker's headphones; a second alert is given once the transaction has been successfully collected. This takes less than 500 milliseconds. Once the transaction has been captured the app stores the transaction data for transmission to the rogue merchant later. As soon as the app has collected a transaction, it automatically returns to waiting to detect another EMV card; it is now ready to collect the next transaction.

Figure 34 shows the data fields as captured by the app, this includes all of the data and cryptographic authorisation codes required by the bank to accept the transaction as genuine.

The mobile app stores transaction data until it has an Internet connection, at which point the app transmits the data to the rogue merchant.

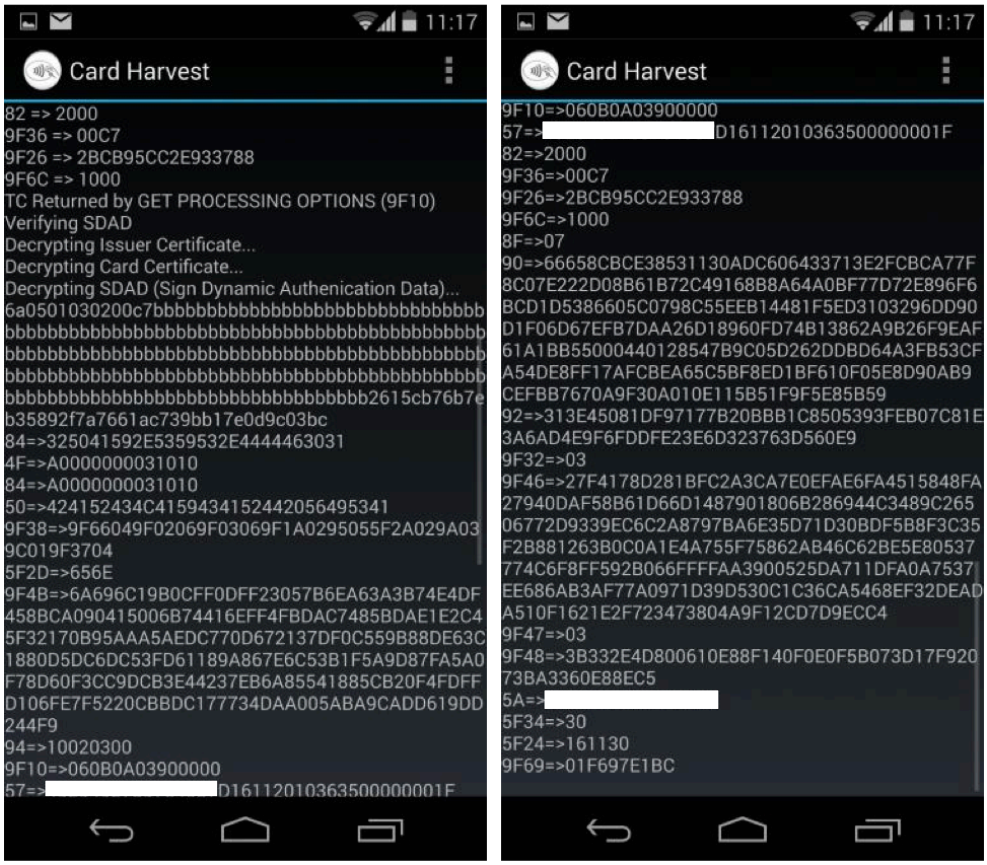


Figure 34 - Captured Transaction Data

The code implements the kernel 3 fDDA [7] contactless transaction protocol sequence, Figure 18, as this is an *offline* only contactless protocol. This allows the attack to be performed in less than 500 milliseconds and avoids additional validation by the bank.

The transaction data is sent by the card in TAG / Length / Value (TLV) format. The Android application stores all of the data fields returned by the card for later transmission to the rogue merchant.

Our software can collect and store multiple offline transactions, without a connection to the Internet. The stored transactions can then be transmitted once a suitable connection is available. The transaction details will include all of the data fields required by the bank. The Application Cryptogram (AC) and the clear text equivalent fields listed in section 3.5.2 are arguably the most important, as together they are used by the bank to verify and thereby approve the transaction.

6.6.3 The Rogue Merchant Application

The rogue merchant application consists of three processes:

- an Internet listening service to receive the captured transactions from the Android app

- a data conversion module which converts the EMV data in TLV format into the ISO 8583 / Standard 70 format required by the Issuing bank
- a POS terminal emulation which sends the formatted data to the bank to collect the money from the fraudulent transactions

6.6.3.1 Internet Based Listening Service

This is a simple Internet based service which listens to a pre-agreed IP address and port number. The Android transaction capture app connects to the pre-arranged IP address and port number to send all of the collected transactions to the rogue merchant. The listening service stores the transactions for later processing.

6.6.3.2 Data Conversion Process

The data conversion process accepts TLV data as captured from the EMV credit / debit card and converts it into ISO8583 / Standard 70 format required by the bank.

To request the money from the victim's account, the rogue merchant must send a financial presentment message (in ISO8583 or Standard 70 format) to the acquirer bank that holds their merchant account.

Table 7 shows the data fields required by the ISO 8583 financial presentment message and shows how the rogue merchant will complete the data fields from the data generated by the EMV card during transaction approval.

6.6.3.3 POS Terminal Emulation

Once the financial presentment request message has been generated, it is sent to the acquirer bank to complete the transaction and transfer the money from the victim's bank account into the rogue merchant's account.

In the UK, communications with the acquirer bank over a public IP network must be protected using Secure Sockets Layer/Transport Layer Security (SSL/TLS) or IPSec [65].

The use of standard encryption such as SSL/TLS and/or IPSec allows the rogue terminal to be implemented in Java on a PC platform; no specialist hardware is required, illustrated in Table 7.

Table 7 - Financial Presentment Message Data Requirements

Item	Name	Description and mapping to EMV card data
1	bit map extended	List of fields included in the message
2	primary account number	***0x5A** – 16-digit card account number
3	processing code	Constant 00 for goods and purchases
4	amount, transaction	**0x9F02** – the transaction amount

5	amount, reconciliation	Transaction amount **0x9F02** converted into the currency to be applied to the victim's card, this value is calculated by the rogue POS terminal
7	date and time, transmission	Date and time the rogue POS transmits the transaction to the bank
9	conversion rate, reconciliation	Conversion rate for the reconciliation amount, calculated by the rogue POS terminal
10	conversion rate, cardholder billing	As above; this value is calculated by the rogue POS terminal
11	systems trace audit number	Transaction sequence number generated by the rogue POS terminal
14	date, expiration	**0x5F24** – Expiry date of the card (YYMM)
16	date, conversion	Date / time of the currency conversion (same as 7)
19	country code, acquiring institution	Country code of the rogue POS terminal (e.g. 0826 for UK, 0840 for USA, 0036 for Australia)
20	country code, primary account number	**0x5F28** – Country code for the card i.e. 0826 – UK
21	country code, forwarding institution	**0x5F28** – Country code for the bank that issued the card i.e. 0826 – UK
22	point of service entry mode	Type of POS terminal, constant value "051" for Chip & PIN / EMV contactless terminals
23	card sequence number	**0x5F34** – Identifies subsidiary EMV cards issued on the same 16-digit account number
25	point of service condition code	Constant "00" normal card presentment
26	point of service PIN capture code	Constant "x8xx" indicates a POS terminal that accepts up to 8 digits
27	approval code length	Constant set by acquirer bank
32	acquiring institution identification code	Constant set by acquirer bank

33	forwarding institution identification code	Constant set by acquirer bank, indicates the institution that will provide the card payment clearing (steps 6 to 9 in Figure 31)
34	primary account number, extended	Not applicable to kernel 3 – used only when the primary account number begins with "59"
39	action code (was response code)	Constant "0xx" for financial transaction request messages
43	card acceptor name/location	Constant string name and location of the merchant
49	currency code, transaction	**0x5F2A** – Transaction currency code
50	currency code, reconciliation	Currency code for reconciliation, see item 5
51	currency code, cardholder billing	**0x9F42** – Currency Code from the card.
66	country code, receiving institution	**0x5F28** – Country code for the bank that issued the card i.e. 0826 – UK
100	receiving institution identification code	Code that identifies victim's bank – ISO 7812
102	account identification 1	Information contained in 16-digit card account number **0x5A**
103	account identification 2	Information contained in 16-digit card account number **0x5A**

*EMV data fields from the EMV protocol are denoted by their EMV reference number e.g. **0x5A**.

Table 8 shows the communication sequence required for the POS emulation to transmit a transaction to the acquirer bank.

Table 8 - POS / Acquirer Communication Sequence

Message	From → To	Purpose
financial presentment request message	POS → Acquirer	Requests approval and money transfer by the acquirer
financial presentment response	Acquirer → POS	Contains the answer to the request
financial presentment confirmation	POS → Acquirer	Confirms that the response was received

6.7 Test Results

The attack software has been tested against various UK-issued credit / debit cards. Table 9 shows the vulnerability of several different card types.

Table 9 - Vulnerability of UK-Issued Contactless Card Types

Card Type	Max Value	Comment
Kernel 3 credit cards (UK currency)	85.00	Kernel 3 credit cards will approve multiple transactions until offline limit reached
Kernel 3 credit cards (foreign currency)	999,999.99	Kernel 3 credit cards will approve foreign currency transactions up to the maximum value possible in EMV
Kernel 3 debit cards (UK currency)	45.00	Kernel 3 debit cards will approve multiple transactions until offline limit reached
Kernel 3 debit cards (foreign currency)	0.00 to 5,000.00	The value authorised by kernel 3 debit cards varied dependent on Issuing Bank
Kernel 2 credit and debit cards	Not Applicable	Kernel 2 (MasterCard) is not affected by this attack as the cards request online completion of transactions in local currency and foreign currencies

6.7.1 Transaction Capture Timings

The Android transaction capture app is designed to operate as quickly as possible, thereby reducing the risk of detection for the attacker. The software automatically collects the fraudulent transaction as soon as it detects a Kernel 3 contactless credit or debit card. Table 10 shows analysis of protocol timings from 20 captured fraudulent transactions.

Table 10 - Fraudulent Transaction Capture Timings

Statistics	Time (ms)
Average transaction duration (card discovery to transaction approval)	478
Standard deviation	36
Fastest transaction	452
Slowest transaction	527

6.8 Potential Solutions

The key weakness exploited in this chapter is that Kernel 3 credit cards will authorise unlimited value transactions in a foreign currency. This makes the attack described in this chapter both scalable and very lucrative.

The solution is relatively simple. This can be done by changing future Kernel 3 credit cards to implement one or both of the following:

- the cards will request *online* completion of contactless foreign currency transactions; making the transaction subject to the additional *online* verification steps.
- the cards will force Chip & PIN completion of all foreign currency transactions; this will eliminate the possibility of high value transactions without the added security of cardholder's PIN.

6.9 Conclusion

In this chapter we have demonstrated that it is possible to collect high value transactions from contactless Kernel 3 credit cards whilst the card is still in the victim's wallet. The attack exploits a previously undocumented flaw in the cards, in which the cards will approve transactions of unlimited value in a foreign currency. Combined with the lack of POS terminal authentication and the threat of contactless payment card skimming, this vulnerability poses a real risk that allows high value fraudulent transaction to be harvested and converted into money.

Our experimental results show that the attack could be implemented in the "real world", as the average time to capture the transaction was less than 500 milliseconds and Android phones with NFC are cheap and readily available

We have also outlined a scenario by which the captured fraudulent transactions could be exploited by a rogue merchant to access the money in the victim's bank account. The rogue merchant receives the transactions and passes them off as genuine transactions to their bank. It should be noted that although we have implemented the rogue POS terminal software, we have not tested it against a live acquirer transaction clearing system.

We have proposed two simple changes in the operation of Kernel 3 credit cards that would eliminate the risk posed by this attack. Both of which use the existing functionality of the cards and would therefore be relatively inexpensive to implement.

Chapter 7. Risks of Contactless Verify PIN

This chapter describes a vulnerability discovered by using the analysis methodology described in Chapter 5. The vulnerability arises from an omission in the kernel 3 (Visa) contactless specification. Whilst the kernel 2 (MasterCard) specification [67] defines that the functionality of 'offline contactless PIN and Verify' is excluded for security reasons, the kernel 3 card does not have an equivalent statement. This omission leaves it unclear what the recommended action should be for Kernel 3 cards when they receive a contactless PIN Verify command.

Because the specification does not say what to do, this leaves the implementation decision in the hands of the card manufacturer and the Issuing Bank. Some Kernel 3 contactless cards excluded the functionality. However, as at 2014, 18.5 million [31] UK issued kernel 3 contactless cards did include the PIN Verify functionality.

7.1 The Contactless PIN Verify Vulnerability

In the UK the EMV specification for contact transactions supports PIN verification locally by the card (*offline*) and PIN verification remotely by the bank's computers (*online*). The specifications for contactless transactions specifically exclude the use of *offline* PIN verification (full details in [7] Book A section 5.9.3 and [68] section 2.4 point 5) because contactless *offline* PIN verification would require the PIN to be transmitted wirelessly to the card which poses a security risk from eavesdropping.

The EMV specification does not permit PIN entry for contactless cards. PIN entry is allowed on transactions made using NFC enabled mobile devices. Mobile device payments are controlled by Consumer Device CVM[2] rules, which permit *online* PIN verification, but not *offline* PIN (full details in [7] Book C3 sections 2.1 and 5.7).

This chapter examines the security implications of the verify PIN functionality intended for Chip & PIN operation also being available over the contactless interface, where it can be accessed without the

[2] Cardholder Verification Method (CVM) is the method by which the cardholder proves that they are the genuine cardholder and the PIN or by signature

cardholder's knowledge or consent. Surprisingly many of the contactless cards currently in circulation in the UK allow access to *offline* verify PIN.

The attack scenario presented draws upon research carried out into the predictability of PINs [69] which shows that there is a subset of PINs that are much more commonly used; meaning guesses from this subset are much more likely to be successful.

The implementation work builds upon related investigations into the vulnerability of EMV contactless payment cards to various attacks, such as skimming [51] [41] and transaction relay [43] [42] . These papers show that the wireless interface makes contactless payment cards vulnerable to new modes of attack that were not present in Chip & PIN. Other research [70] [13] show that the EMV protocol sequence can be manipulated to produce erroneous behaviour in the cards and the POS terminals.

In what follows, we first introduce the attack scenario then the technology used and finally the performance results demonstrating the practicality of the attack. A critical part of our software implementation is the ability to find and attack EMV payment cards contained in a wallet with various other contactless cards. Our software implements the ISO-14443 part 3 protocol sequence for card initialisation and anti-collision. It can identify multiple cards, select each card in turn and communicate with each card once selected.

7.2 Attack Scenario

The attack scenario outlined in this chapter is presented as supporting evidence of our assertion that allowing contactless access to *offline* verify PIN represents a tangible threat to a large number of EMV payment cards currently in circulation in the UK.

Newcastle University, like many other companies and institutions, uses NFC enabled identity cards to control access to our buildings. When entering the building, many of us place our whole wallet on the door access reader as it is quicker and easier than taking the access card out of the wallet. This gives an attacker the opportunity to access the other cards in the wallet, communicating with any contactless payment cards also present.

Given that the person will enter the building on a regular basis and that the number of available PIN attempts is reset each time the payment card is used in a POS terminal or ATM, the attacker can have unlimited attempts to guess a card's PIN.

In our experimental implementation of the attack scenario we make use of (i) a protocol sequence which exploits the verify PIN functionality (ii) the ability to access multiple cards in a single wallet presented to the door access reader (iii) a strategy for guessing PINs [69] which will yield the greatest number of correct guesses.

7.2.1 PIN Verify Protocol Sequence

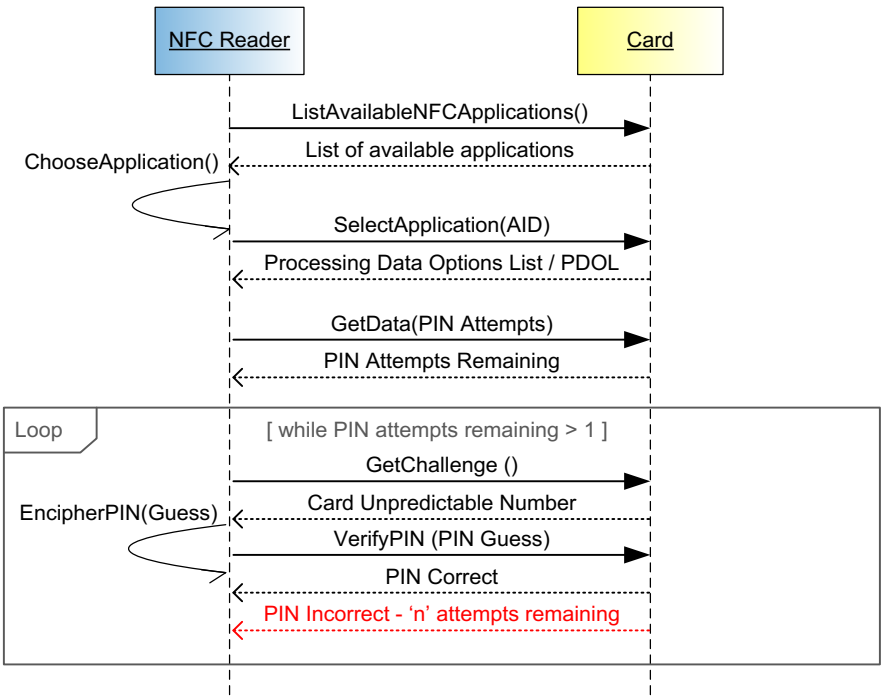


Figure 35 - Verify PIN Protocol Sequence

The full protocol sequence, Figure 35, is designed to guess the PIN without locking the card; locking occurs when all of the available PIN attempts are used (i.e. PIN attempts remaining > 1). The Verify Pin protocol uses the minimum number of commands possible so that it can be completed quickly (<500ms) to avoid arousing the suspicions of the cardholder. The protocol sequence is limited to a maximum of two guesses each time the cardholder uses the door. However, over time the attack has multiple chances to run the protocol sequence as the person will regularly return to the door access reader and each time the card is used in a POS terminal or ATM, the PIN attempt counter is reset giving more chances.

The PIN verify protocol sequence described above ensures that at least one PIN attempt is left on the card. However, the logic can be changed to create a nuisance attack which wipes out all of the available PIN attempts on all of the EMV payment cards in the wallet. This would not yield any financial gain, but there are many malicious attacks performed purely for the nuisance value. A card that has zero PIN attempts remaining cannot be reactivated at the POS terminal and the cardholder must go to a bank ATM.

7.2.2 Reading Multiple Cards

The scenario requires reader software capable of distinguishing between multiple NFC cards in a wallet, allowing it to locate the EMV payment cards (implementation details can be found in section 7.3.1). This also gives the potential to look for additional data such as the cardholder's birthday on

the other cards in the wallet, such as loyalty cards which may hold personal data unencrypted. Bonneau et al. [69] shows that knowing the person's birthday increases the chances of guessing their PIN within 6 guesses from 1.94% to 8.23%.

7.2.3 PIN Guessing Strategy

The attack scenario presented accesses the card each time the cardholder enters the building. This gives it potentially unlimited guesses at the PIN over time, two guesses each time the door access is used. Bonneau et al. [69] presents a survey containing a study of 1,351 respondents, 805 of which detailed the respondents choice of PIN and their reason for choosing it. The survey shows that 23% of respondents chose a memorable date (birthday and anniversary) as their PIN. The paper goes further and identifies a list of PINs which are statistically more likely; using this list, the paper calculates that given 6 guesses, the chance of correctly guessing the PIN is 1.94%, which rises to 8.23% if the birthday of the cardholder is known. This research is backed up by a recent news story [71] where a burglar stole a wallet in which he found a driving licence and two ATM cards, he correctly guessed the PIN from the date of birth on the driving licence and was able to obtain £1,000 from a nearby ATM.

7.3 Software Implementation

The experimental work in preparing this chapter includes (i) an implementation of the verify PIN protocol sequence which makes multiple attempts to guess the PIN of any EMV payment card detected in the wallet (ii) a multiple card reader implementation which will identify and communicate with all of the contactless cards in the wallet.

The experiments were performed using an ACR122-U contactless card reader [72] and the Java™ Smart Card I/O API [73].

7.3.1 Verify PIN Implementation

The UML sequence diagram, Figure 35, illustrates the protocol sequence required to perform the verify PIN attack sequence. The sequence employs the minimum number of commands which achieve two contactless verify PIN attempts, this minimises total execution time (on average 457.2ms) for the sequence. Minimising execution time is important to ensure that the attack is not easily detected by the cardholders using the door access.

The protocol sequence is initiated when the multiple card reader, section 9.2.1, detects an EMV payment card in the wallet. The protocols sequence therefore starts with the EMV payment card in the `active` state ready to accept commands, see Table 16 for a full explanation of the possible card states. Once the reader has established communication with the card it reads the number of PIN attempts remaining using `GetData(PIN Attempts)`. It then calls the verify PIN command in a loop. The card responds with 0x9000 if the PIN is correct or 0x63Cn if the PIN is incorrect, where 'n' is the

number of PIN attempts remaining. The loop is repeated until the correct PIN is guessed or only one PIN attempt remains.

We observed that the contactless PIN is the same as the contact PIN, this was confirmed by changing the card's contact PIN using an ATM and verifying that the contactless PIN had also changed

7.3.2 Verify PIN Protocol Sequence

Based on the data obtained in our tests the average time required to perform the PIN verify protocol sequence, Figure 35, was only 457.2ms; thereby strengthening the case that the door access reader attack scenario can be implemented without raising the suspicions of the users of the door access system.

The time taken to perform each of the commands in the verify PIN protocol sequence is detailed in Table 11 which shows the average time and standard deviation calculated from 20 test runs performed using EMV payment cards issued by a UK bank.

Table 11 - Verify PIN Command Execution Times

Command	average (ms)	Std dev (ms)
`ListAvailableNFCApplications()`	18.4	12.7
`SelectApplication(AID)`	19.2	5.5
`GetData(PIN attempts)`	29.8	17.9
`GetChallenge()`	24.6	7.0
`VerifyPIN(incorrect PIN)`	175.8	7.2
`GetChallenge()`	12.2	6.8
`VerifyPIN(correct PIN)`	177.2	9.6
`Complete Protocol Sequence`	457.2	24.9

The results show that 77.2% of the total time was taken by the card responding to the VerifyPIN() command. It is also interesting to note that there is no significant difference between a correct PIN (177.2ms) and an incorrect PIN (175.8ms).

7.4 Summary

The attack scenario described in this chapter exploits contactless verify PIN to give potentially unlimited attempts to guess the cardholder's PIN without their knowledge. The implementation work has successfully built and tested software that proves this attack scenario is technically viable. The timing tests prove that the attack protocol sequence can be performed in less than 1 second, making it possible to access the payment cards in the wallet without arousing the suspicions of the cardholder. In the suggested scenario the attack protocol sequence can access a victim's card multiple times over a number of days, this significantly increases the odds that the attack will guess their PIN correctly.

It is our assertion that the attack scenario and experimental implementation work presented in this chapter make a compelling case that contactless verify PIN can be misused in order to find out the PIN of the card without the knowledge of the cardholder. This significantly impacts the underlying security assumption of the Chip & PIN payment system, that an attacker can only gain knowledge of the cardholder's PIN through the negligence or collaboration of the cardholder. Moreover, offline verify PIN is not required in the processing of contactless transactions and is therefore a redundant functionality. These findings suggest that it would be prudent to remove the contactless verify PIN functionality. It may also help if cardholders are educated about the risks of placing their whole wallet on a contactless reader rather than removing the card from the wallet prior to use.

7.4.1 Verification by Testing and Analysis

Testing was performed using the specific protocol sequences developed for each card, the results of the tests were as follows

- Kernel 2 contactless credit and debit cards from several different UK banks were tested, all of the cards responded with an error code "command not available".
- Kernel 4 payment cards all responded with an error code "command not available".
- Kernel 3 contactless debit cards from a number of UK banks were tested which were based on the NXP Smart MX chipset, these cards responded with an error code "command not available".
- Kernel 3 credit and debit cards issued by a major UK bank, responded with PIN OK / PIN INCORRECT. Indicating that the offline PIN verify command was accessible via the contactless interface. It was noted that all of the cards that responded in this way were based on an Infineon Chipset. Corporate contactless cards issued by the same bank, with the NXP Smart MX chipset, were not affected by the verify PIN flaw.

The conclusion that can be drawn from the experiments is that this is not a fundamental error in the EMV contactless specification, it is rather, an error in interpretation of the specification by a given UK bank and the card manufacturer.

However, the error can be traced back to the specification being too ambiguous. Kernel 2 clearly states offline PIN Verify is "not allowed". The kernel 3 and kernel 4 specifications do not specifically state either way, and the general case for all card types given in [7] Book A only states offline PIN Verify is not suitable due to timing issues.

EMV cards and POS terminals are implemented with maximum interoperability in mind, with the aim of never declining a transaction because the card and the POS terminal are not compatible. The specification should always clearly state when functionality is undesirable for security reasons as the default assumption will always be to include functionality that is not necessarily required for future compatibility.

7.5 Conclusion

This vulnerability shows that the contactless interface can allow unauthorised access to the secure contact Chip & PIN functionality of the card. Therefore it can be argued that this shows that a usability feature of contactless payments impacts on the security of EMV Chip & PIN cards.

As part of the responsible disclosure of this discovery several months before this chapter was published at Financial Cryptography and Data Security 2013, we have shared our findings and the test results to the UK Cards Association, Visa, MasterCard and the UK bank whose cards were affected.

This led to a full discussion with the UK bank about how the problem with offline PIN verify was discovered and we provided the bank with the details of the results from the different card chipsets NXP and Infineon.

Chapter 8. POS Authentication

Several University research teams have demonstrated that the wireless interface makes EMV contactless cards vulnerable to new types of attack that were not possible for EMV Chip & PIN cards: skimming attacks [41], eavesdropping attacks [50], access the card's secure functionality [27] and harvesting fraudulent transactions, Chapter 6. These attacks are possible because the cards will communicate with any device equipped with an NFC reader that comes within range, even without the cardholder's knowledge. To address this vulnerability, this chapter presents a POS terminal authentication solution for EMV contactless payment cards. Terminal authentication allows contactless cards to distinguish genuine POS terminals from other NFC readers and thereby restrict the access to its sensitive data and secure application functionality.

The POS Authentication work was carried out in collaboration with Joseph Hannon. We have implemented and tested a prototype transaction protocol. Our experiment shows that it was possible for the card to successfully authenticate the POS terminal and for the POS to authenticate the card. This was achieved using the existing kernel 3 (Visa) contactless transaction message sequence, Figure 18. We implemented our prototype transaction protocol with an Android mobile phone acting as the payment device and a PC acting as the POS terminal.

The challenge was to create a method of preventing EMV cards from revealing sensitive data to unauthorised readers. This must be performed within the constraints of

- completing the transaction in less than 1 second to make it compatible with a consumers expectation for a contactless transaction
- perform an additional cryptographic authentication process whilst keeping data volume within current EMV restrictions of 256 bytes
- maintain the cryptographic security of transaction
- compatibility with the existing EMV infrastructure

Our solution combines several existing technologies in an innovative way to solve the complex issues related to adding functionality to a global payment system such as EMV. The contribution of the solution is outlined below.

8.1 Outline of Proposed Solution

The Chip & PIN technology on EMV payment cards have been proven relatively secure against card fraud since the introduction of DDA and CDA cards in 2009. Contactless payment technology adds a wireless interface to EMV cards, which allows it to be accessed whilst it is still the cardholder's wallet, even without their knowledge.

In the current EMV payment system the cards must prove to POS terminals that they are genuine, the reverse is not true. NFC enabled devices such as smart phones are becoming more common, these can mimic POS terminal's behaviour. The lack of terminal authentication could lead to cases where rogue terminals might carry out skimming and eavesdropping attacks, leading to fraudulent transactions.

Our solution aims to address this imbalance by making it necessary for *any* NFC enabled device attempting to access the card to authenticate itself. In other words, our solution will prevent the card from revealing sensitive information, unless the NFC reader can prove it is a genuine POS terminal issued by a bank.

Any NFC reader attempting to access the card will be challenged by the card to authenticate itself as a genuine bank-issued POS terminal. The card sends an unpredictable number (nonce) to the POS terminal, and the terminal must authenticate itself by producing an Elliptic Curve Digital Signature Algorithm (ECDSA) signature based on this nonce challenge Figure 39. The card is able to validate the ECDSA signature, Figure 38, using a three-tier Public Key Infrastructure (PKI) based on the Certificate Authority public key. Once POS authentication is complete the transaction can continue and card authentication is performed as per the current EMV protocol.

Restricting Access to the Card's Sensitive Data / Secure Functionality.

The card will restrict access to sensitive data and secure functionality until POS authentication has taken place. This is achieved using a state machine similar to that currently implemented by MasterCard [68]. The state machine, Figure 37, will control the sequence in which EMV commands can be called, and thereby the data that can be accessed.

Integration with the Existing EMV Protocol Sequence.

There are seven EMV contactless protocol sequences, kernel 1 - Visa, kernel 2 - MasterCard, kernel 3 - Visa fDDA, kernel 4 - American Express, kernel 5 - JCB, kernel 6 - Discover and kernel 7 – UnionPay. The POS authentication functionality has been designed so that it can be incorporated into all seven protocol sequences without changing the command sequence. To achieve this, the solution adds new data fields, Table 13, to the two commands that occur at the start of all seven protocol sequences: `Select()` and `GetProcessingOptions()`. Figure 36 shows how the POS authentication functionality is incorporated into the kernel 3 protocol sequence.

Elliptic Curve Cryptography.

The EMV protocol has a restricted message size of 256 bytes. This does not cause problems in the existing RSA authentication of the card by the POS terminal, as the card can pass several 256-byte messages in response to a single POS terminal message. However, this does cause a problem in the proposed POS authentication solution as the POS terminal must pass all of authentication information in a single 256-byte message. The solution for this problem is to use Elliptic Curve Cryptography (ECC), which provides increased cryptographic strength over the existing RSA scheme [74] whilst allowing the authentication information to be passed in a single (256 byte) message. The proposed ECC curve and algorithms are compliant with EMV's proposed adoption of ECC [75].

8.2 Transaction protocol

There are seven variations of the EMV contactless transaction protocol sequence. The prototype implements the kernel 3 (Visa fDDA) transaction, Figure 18. Kernel 3 was selected as it contains the least number of commands of any of the seven protocol sequences, and therefore it was the most challenging protocol for incorporation of the POS authentication functionality. Given that we can incorporate the POS authentication protocol into kernel 3 we should therefore be able to incorporate POS authentication into any of the other kernels.

In Figure 36, the additions to the protocol required for POS authentication are coloured blue:

- POS authentication uses the `Select()` command, Figure 36 - point 6.0, and the `GetProcessingOptions()` command, Figure 36 - point 8.0, since these command are common to all seven contactless protocols. `Select()` and `GetProcessingOptions()` are also the first two commands in each of the protocol sequences which allow the protocol sequence to be halted before any sensitive data is divulged by the card.
- The request for POS authentication has been added to the PDOL returned by the `Select()` command, Figure 36 - point 6.3. Details of the new structure of the PDOL are given in section 8.2.1. The PDOL was chosen as the trigger for POS authentication as it is currently the way that the card requests information from the POS terminal.
- The nonce is also contained in the response to the `Select()` command, Figure 36 - point 6.3. The nonce is an 8-byte unpredictable number which the POS terminal must sign with its private key to produce the ECDSA authentication signature, Figure 38 - ECDSA Signed Nonce. A nonce is used to ensure that the ECDSA signature cannot be recorded by an attacker and replayed to gain access to the card.

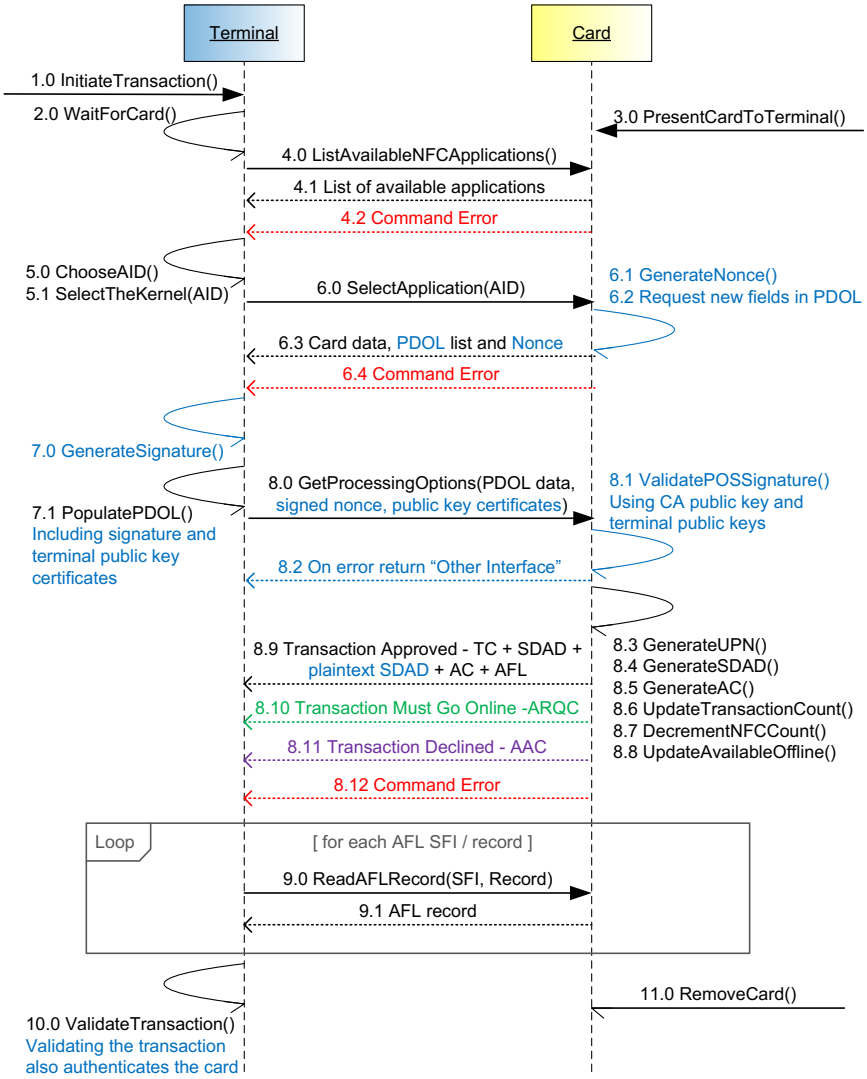


Figure 36 - Prototype POS Authentication Transaction protocol Sequence

The POS terminal generates the ECDSA authentication signature, Figure 36 - point 7.0, which is then returned in the message data contained in the `GetProcessingOptions()` command, Figure 36 - point 8.0. The message also contains the Elliptic Curve Qu-Vanstone (ECQV) implicit certificates and other data required by the card to validate the ECDSA signature.

If POS authentication fails, the card returns "Try Another Interface", Figure 36 - point 8.2, which will cause the POS terminal to request a Chip & PIN contact transaction. This has the advantage that it ensures that the transaction is not lost if the POS terminal is not compatible with POS authentication. POS terminals which have not been updated to include POS authentication will follow the existing EMV protocol and continue the transaction in Chip & PIN mode [7]

8.2.1 Processing Options Data Object List (PDOL)

The PDOL is a list of data fields that the card requests from the POS terminal. For POS authentication, the PDOL must contain all of the standard transaction fields, Table 12. In addition, it must also contain the new fields required for POS authentication, Table 13.

Table 12 - Transaction Data Fields in the Current kernel 3 PDOL

TAG	Length	Description
9F66	4 bytes	Terminal Transaction Qualifiers (kernel 3 specific; this tag will need be changed for each of the different issuer card types)
9F02	6 bytes	Transaction amount
9F03	6 bytes	Amount other (used for cashback always zero)
9F1A	2 bytes	Terminal country code
95	5 bytes	Terminal verification results (always zero at this stage)
5F2A	2 bytes	Transaction currency code
9A	3 bytes	Transaction date
9C	1 bytes	Transaction type (always 00 for purchase)
9F37	4 bytes	POS terminal nonce

Table 13 - Fields Required for POS Authentication

TAG	Length	Description
9F81	64 bytes	ECDSA signed nonce
9F82	64 bytes	Acquirer implicit certificate (Figure 38 - Aic)
9F83	9 bytes	Acquirer ID data
9F84	64 bytes	Terminal implicit certificate (Figure 38- Tic)
9F85	9 bytes	Terminal ID data (Figure 38- Tid)

8.2.2 Acquirer ID (Aid) and Terminal ID (Tid) Information

The Aid and Tid contain the plain-text data which is used to calculate the ECQV Acquirer implicit certificate (Aic) and Terminal implicit certificate (Tic). Details of the data contained in Aid and Tid are given in Table 14

Table 14 - Acquirer and Terminal ID Information

Field	Length	Data
ID number	4 bytes	Identity of the acquirer (Figure 38 – Aid) Identity of the terminal (Figure 38 – Tid)
Expiry Date	2 bytes	MMYY
Serial Number	3 bytes	Version number of implicit certificate
Total	9 bytes	

The format of the Aid and Tid data has been designed to follow the issuer information contained in the current EMV RSA public key certificates [6] (i.e. 4-byte issuer ID, 2-byte expiry date, and 3-byte serial number).

8.2.3 Controlling Access to the Card's Data and Functionality

The cards will require a state machine similar to that currently implemented by MasterCard, Figure 37. The MasterCard state machine is relevant to the proposed solution in that the two major control points used there are the `Select()` and `GetProcessingOptions()` commands.

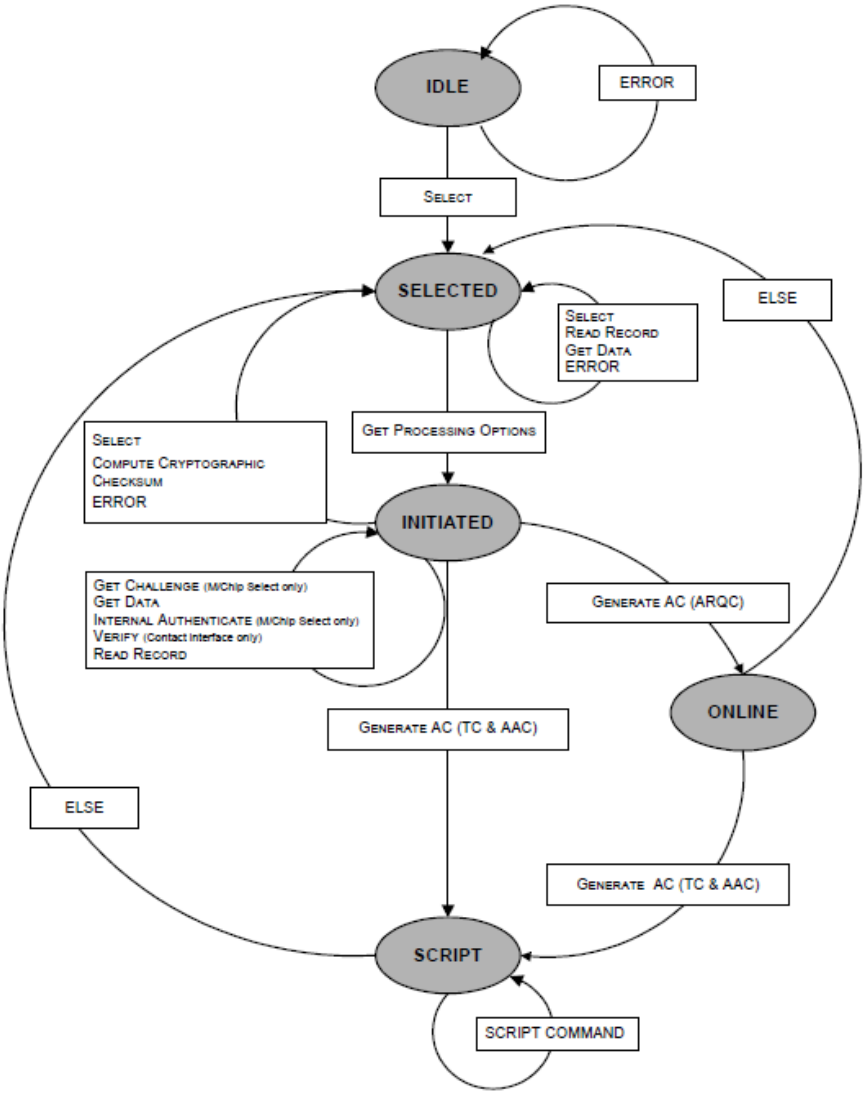


Figure 37 - MasterCard State Machine (source [17])

When the card is "IDLE", `Select()` is the only command that can be called. Successful completion of `Select()` places the card into the "SELECTED" state.

Being in the "SELECTED" state allows access to `GetProcessingOptions()` which puts the card into the "INITIATED" state. In turn, this state gives full access to the card.

In order to make our proposed solution work, there is a need to alter the existing state machine, whereby the `ReadRecord()` command is moved to after the "INITIATED" state, which prevents the card's sensitive information from being read prior to POS authentication being completed.

8.2.4 Elliptic Curve Cryptography (ECC)

A 512-bit ECC scheme has been selected for the proposed solution as it provides a higher cryptographic bit-strength than EMV's current 1984-bit RSA scheme. The shorter ECC scheme also allows the extra data required for POS authentication to fit into the `GetProcessingOptions()` PDOL message.

The solution utilises two elliptic curve schemes, ECQV and ECDSA, which perform different tasks in the implementation. The CA public key stored on the card and the two ECQV implicit certificates supplied by the POS terminal form a three-tier Public Key Infrastructure (PKI), which is used to verify the ECDSA digital signature generated by the POS terminal, Figure 38. Brown et al., 2011 [76] describes a scheme for ECQV-certified ECDSA such as the one implemented in the prototype.

With ECQV implicit certificates, the CA public key is used to generate a public key from the implicit certificate, which in turn is used to generate the public key from the implicit certificate on the subsequent tier of the PKI [77].

8.2.5 Elliptic Curve POS Authentication Process

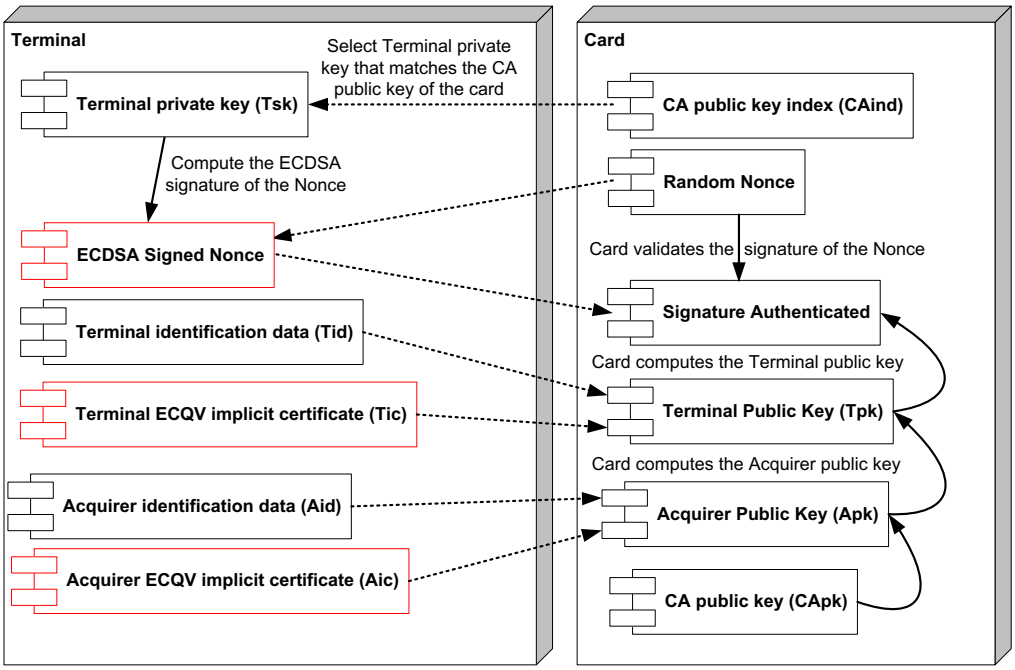


Figure 38 - POS Authentication by Card

The POS terminal authenticates itself by creating an ECDSA signature of the nonce produced by the card, Figure 38, using its private key (Tsk). The card validates the ECDSA signature using the CA public key (CApk) and the two ECQV implicit certificates: the Acquirer (Aic) and Terminal (Tic). The card generates the Acquirer public key (Apk) and Terminal public key (Tpk) from the ECQV

implicit certificates (Aic) and (Tic). The card authenticates ECDSA signature using the Acquirer public key (Apk) and Terminal public key (Tpk) and the card's copy of the nonce produced by the card.

The solution depends on the POS terminal signing a nonce generated by the card; the nonce ensures that the ECDSA signature is fresh each time the card requests authentication. The three-tier Public Key Infrastructure (PKI) links the ECDSA signature produced by the POS to the CA public key (CApk), which proves that the Terminal's private key was issued by an authorised Acquirer bank.

The POS terminals keys will be stored on the Secure Access Module (SAM), the SAM protects the keys from being read by brute force. The SAM also allows additional cryptographic functionality to be added to the POS without requiring an upgrade. Distributing the keys on the SAM avoids transmitting the keys to the POS terminal and thereby risking interception.

8.2.6 Elliptic Curve Generation of POS Terminal Keys

To perform the POS authentication process detailed in Figure 38 the POS terminal must have two ECQV implicit certificates: the Acquirer implicit certificate (Aic) and the Terminal implicit certificate (Tic), as well as a Terminal private key (Tsk). These are generated in the process detailed in Figure 39.

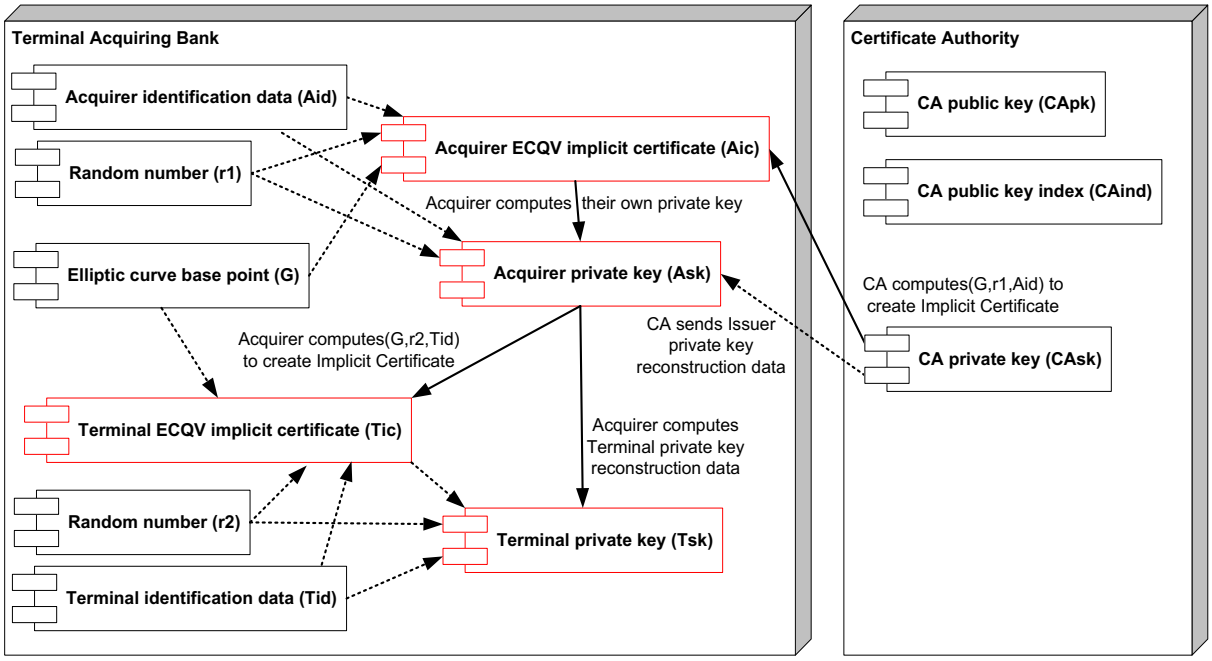


Figure 39 - Generation of POS Terminal Keys

Figure 39 shows that the Acquirer implicit certificate (Aic) and Acquirer private key (Ask) are generated by the Certificate Authority based on the data supplied by the Acquiring bank. The Acquiring bank generates the Terminal implicit certificate (Tic) and Terminal private key (Tsk) using the Acquirer keys (Aic) and (Ask).

The Certificate Authority, Acquirer and Terminal keys form a three-tier Public Key Infrastructure (PKI) that enables the card to validate the ECDSA digital signature produced by the POS terminal based on the Certificate Authority public key (CApk) stored on the card.

In the current EMV PKI, there is a CApk for each of the six card issuers (Visa, MasterCard, American Express, JCB, Discover and UnionPay). The POS terminal will have to store a set of keys (CApk, Aic, Tic and Tsk) for each card issuer that the POS terminal wants to accept.

8.2.7 Elliptic Curve Card Authentication

Based on the current EMV specification, the card authenticates itself by producing an RSA signature of the transaction data, referred to as the Signed Dynamic Application Data (SDAD), Figure 36 - points 8.4 and 8.9. The POS terminal validates the SDAD RSA signature and thereby the card, Figure 36 - point 10. In the proposed solution, the SDAD has been altered to be an ECDSA signature rather than RSA. This alteration means that all of the cryptography used in the proposed solution is based on elliptic curve.

8.3 Prototype Implementation

The prototype implementation consists of a prototype POS terminal and a prototype payment device. The payment device is a mobile phone emulation of a kernel 3 contactless card which incorporates the POS authentication functionality.

8.3.1 Prototype Payment Device

The payment device has been implemented as a card emulation on a Nexus S Android mobile phone. Implementation on Java Card would have been preferred, however, this was not practical since Java Card does not support ECQV natively using the card's cryptographic co-processor. The current version of Java Card 2.2.2 does support ECDSA and Elliptic Curve Diffie-Hellman (ECDH) natively [78], it is therefore assumed that EMV payment cards could support ECQV in the future if the demand from the banks was great enough.

The Nexus S Android mobile phone was selected for prototype development as it provides both contactless (NFC) communication and the functionality required to generate the ECQV public key certificates and verify the ECDSA signature.

The ECQV and ECDSA cryptography software for the Android mobile phone platform has been implemented from scratch as the Android SDK does not natively support ECQV implicit certificates. This required the implementation of the methods for elliptic curve point addition and multiplication.

To ensure compatibility with future implementations of EMV, the prototype uses the NIST elliptic curve P-256 defined by EMV in their ECC proposal document [75].

For consistency the payment device application also uses ECDSA to generate the SDAD signature, which the POS terminal uses to authenticate the card.

8.3.2 Prototype POS Terminal

The prototype POS terminal is implemented on a PC with an ACR-122U contactless reader. The POS terminal implements the Kernel 3 fDDA protocol sequence outlined in Figure 36 with some additional data fields detailed in Table 13. The Kernel 3 fDDA protocol was chosen because it is the shortest (i.e. it has the most stringent requirements regarding size) of the contactless protocol sequences and thereby demonstrates that the prototype can be implemented in any of the other contactless protocol sequences.

The prototype POS terminal implements the Kernel 3 fDDA transaction protocol sequence as defined in the EMV Contactless Specification Book C-3 [7]. Should POS authentication fail, the prototype payment device (Android phone) will return a "Try Another Interface" as the transaction outcome (see Book C-3 [7] Section 5.2.2.2). This will cause the POS terminal to initiate Chip & PIN completion of the transaction.

The prototype POS terminal implements ECQV and ECDSA cryptographic functionality required to generate the ECDSA POS authentication signature and verify the SDAD ECDSA signature produced by the prototype payments device.

8.4 Prototype Test Results

We have carried out initial experiments of running the prototype solution using a Nexus S Android mobile phone as a card emulation. For now, we have to use this emulation method because the current Java Card 2.2.2 does not support ECQV. However, it is envisaged that the ECDSA signature verification process will be faster when ECQV support is available on payment cards' cryptographic co-processor.

8.4.1 Prototype Transaction Timings

Table 15 shows a comparison between the average time to complete the current kernel 3 transaction protocol and the average time to complete our prototype bilateral authentication transaction protocol. The detailed breakdown shows each stage of the transaction protocol, as per Fig. 1. Steps shared by the two protocol sequences are shown in black and the new steps added for bilateral authentication are shown in blue.

The timings provided are average timings from five executions of kernel 3 transaction protocol sequence and five of the bilateral authentication. For the timing experiments we used an Android mobile device with a 1.3GHz quad core processor, 1Gb RAM, 16GB Storage, NFC chip PN544 by NXP, running the Cyanogenmod CM10.1.3 version of the Android operating system.

The total transaction time of our bilateral authentication prototype is 1,521 milliseconds, i.e. adding just over a second to the time taken for a typical kernel 3 contactless transaction (431 milliseconds).

Contactless transactions are by definition designed to be fast so in this context 1.5 seconds is not fast enough.

Table 15 – Comparison of Transaction Time Current kernel 3 Contactless vs. New Protocol

Protocol Sequence Activity	Kernel 3 transaction	POS authentication
2PAY message (card and POS)	28ms	28ms
Read Next Record message (card and POS)	11ms	11ms
Select kernel 3 message (card and POS)	21ms	21ms
Get Processing Options message (card and POS)	22ms	22ms
Card validates the issuer public key	--	210ms
Card validates the terminal public key	--	257ms
Card verifies the terminal ECDSA signature	--	527ms
Card constructs the authorisation message (SDAD)	4ms	4ms
Card generates hash of transaction data	16ms	16ms
Card generates RSA signature on SDAD	247ms	--
Card generates ECDSA signature on SDAD	--	343ms
Terminal reads AFL records from card	82ms	82ms
Total transaction time	**431**ms	**1,521**ms

However, our efforts at optimising the code, combined with the understanding that increasing the RAM available on the Android device would give better performance, have allowed us to reduce the overall transaction time significantly from 4 seconds to 1.5 seconds. It would therefore be safe to assume that performance improvements in the software and hardware will make terminal authentication practicable in the foreseeable future. For instance it may also be possible to speed up the ECDSA signature verification process using the technique outlined by Antipa et al., 2005 [79], which claims to speed up the process by 40%, although due to time constraints we were not able to investigate the benefits of their techniques further.

8.4.2 Prototype Payment Device Log File

Verifying the POS authentication ECDSA signature requires a number of calculations to (i) derive the issuer public key from the EQCV implicit certificate; (ii) derive the terminal public key from the EQCV implicit certificate; (iii) calculate the ECDSA POS authentication signature; (iv) compare the calculated ECDSA signature with the POS authentication sent by the POS terminal.

8.5 Technical Issues of Implementation of POS Authentication

The solution is designed to prevent attacks on contactless cards using devices with NFC readers, such as mobile phones. During the process of designing the solution, the following challenges and possible modes of attack on the solution were identified and solved.

8.5.1 Integration with the Existing EMV Infrastructure

Challenge: Globally there are 23.8 million EMV POS terminals and 1.6 billion EMV credit / debit cards [5]. Therefore any change to the EMV protocol has a major financial impact across many organisations. It is therefore essential that any change to EMV can be implemented as a gradual

replacement program rather than an enforced step change. This requires that cards and POS terminals which implement the new functionality must be capable of running in parallel with existing cards and POS terminals for many years.

Solution: The POS authentication solution was designed to deal with the two scenarios without the loss of the transaction or the customer having to use a different card (i) an existing card making a payment at a POS terminal with the new functionality; (ii) a card with the new functionality making a payment at an existing POS terminal. Note: the other two potential scenarios (existing card – existing POS terminal and new card – new POS terminal) do not have any issue here.

In the first scenario, the new POS terminal will perform the existing kernel 3 contactless protocol.

In the second scenario the card will perform the modified kernel 3 protocol with POS authentication. It does this by requesting specific data fields in the PDOL as described in Table 12. An existing POS terminal will not be able to supply the fields required Table 13 and will therefore exit the transaction with transaction outcome of "Try Another Interface" [7]. In all of the contactless protocols, the outcome of "Try Another Interface" causes the POS terminal to request that the transaction is completed in Chip & PIN mode. This allows the transaction to continue with minimum interruption.

8.5.2 Integration with Existing EMV Contactless Protocol Sequence

Challenge: There are 23million POS terminals globally [5] therefore the cost of any change to EMV is very large. The POS authentication process must therefore be incorporated into the existing contactless protocols without changing the command sequence.

Solution: The solution is integrated into the `Select()` and `GetProcessingOptions()` commands which are the first two commands in all seven contactless protocol sequences. In the `Select()` command, the card requests the fields required for POS authentication, Table 13. In `GetProcessingOptions()`, the POS terminal returns the ECDSA signature that authenticates it as genuine.

The solution takes advantage of EMV's existing Data Object List functionality, which allows the card to request a flexible list of data fields from the POS terminal. In other words, we are using mechanisms that are already there, without the necessity of changing the protocol, Figure 18.

To perform POS authentication, it is necessary to return a digital signature and two key certificates in a single 256-byte `GetProcessingOptions()` message. This is not possible under the current EMV scheme (1984-bit RSA). The solution implements a 512-bit ECC scheme which is cryptographically stronger than the current RSA scheme. It allows two ECQV implicit certificates and an ECDSA signature to form a POS authentication to be sent in a single `GetProcessingOptions()` message.

8.5.3 Revocation of POS Terminal Keys

Challenge: The ability to revoke POS terminals keys is an essential part of the design. This is because the cards have no way of communicating with the outside world apart from through a connection with a terminal. This creates a situation where a single compromised POS terminal could result in every EMV card being potentially compromised.

Consider the following two scenarios: (i) a POS terminal's private key was compromised and used to generate ECDSA signatures; (ii) a genuine POS terminal is stolen or misused to generate ECDSA signatures. In both cases, the ECDSA signatures produced will appear to be genuine to every EMV-compliant contactless card, since the card has no means of directly receiving and storing information about revoked POS terminal keys.

Informing every EMV card of the revoked keys is impractical, as the bank's backend servers can only communicate with the card when it is connected to a POS terminal or ATM. In addition, there is insufficient storage on the cards to store the revoked keys.

The keys must therefore be revoked at the POS terminal. However, it is too late for the acquiring bank to send a message to tell the POS to revoke its key once the POS key is compromised or the POS has been stolen.

Solution: The proposed solution forces the POS terminal to regularly request a "stay alive" authorisation message from the acquiring bank's backend servers. The POS terminal will only be allowed to issue a limited number of ECDSA signatures (e.g. 50) before it must request another "stay alive" authorisation. Limiting the number of ECDSA signature produced by a stolen / compromised POS does not entirely prevent contactless attacks but it does prevent large scale attacks, as these would produce unusually high numbers of POS authorisation requests that could be detected by the bank and shut down accordingly.

The "stay alive" requests would be made in between contactless transactions when the POS terminal was inactive thereby not impacting on the speed of a contactless transaction.

To prevent the private keys being read directly from the POS terminal's storage, the proposed solution recommends the storage of the private keys on the SAM which is the current method of providing safe key storage for EMV POS terminals.

The "stay alive" protocol is a workable solution, however, it could lead to legitimate POS terminals being locked out until they receive a new update of ECDSA signatures.

8.5.4 Safe Storage of POS Terminal Keys

Challenge: Given that there is monetary impact associated with the loss of POS terminal keys, it is important to protect them. There are 23.8 million EMV POS terminals in circulation worldwide [5];

you can even buy POS terminals on eBay. This gives criminals easy access to POS terminals from which they could attempt to extract the POS terminal keys.

Solution: The POS keys must therefore be stored in secure storage. Modern POS terminals already have secure storage for their cryptographic keys, in the form of the Secure Access Module (SAM). The proposed solution will make use of SAM to store the keys required for POS authentication.

8.6 Conclusion

Our prototype implementation of the protocol was successful, proving that it was possible to integrate POS authentication into the existing contactless transaction protocol and have the cards recognise genuine POS terminals and ignore NFC readers not issued by an authorised bank.

Our prototype also clearly demonstrated that the protocol was pushing limits of what was possible within the constraints of the existing EMV payment system. The prototype highlighted the following issues:

- ECC gave us much shorter cryptographic signatures and allowed us to fit the required data into EMV's 256byte message size. The drawback of ECC is that it takes significantly more processing time than RSA which was a major contribution to the threefold increase in transaction time 431ms to 1,521ms, this can be clearly seen in the blue rows in Table 15.
- The POS authentication protocol requires the significant change to the software and storage of new cryptographic keys on the POS terminals. This is a major issue as there are 23 million EMV POS terminals worldwide [5] which represents a very large investment in POS terminals by the acquirer banks and merchants.
- Large numbers of POS terminals in countries such as the UK are connected to the internal networks of large supermarket merchants rather than directly to the payments networks. This makes the distribution and revocation of cryptographic keys extremely problematic.

8.6.1 Feedback on the Proposed POS Authentication Protocol

Feedback from academia and feedback from the payments industry made it clear that modification of the existing EMV contactless protocol is not the answer to the current contactless skimming and relay issues. What is required is a radical redesign and in section 10.2 Future Work, I outline a design for a new contactless transaction protocol which could resolve many of these issues.

The POS authentication solution taught us some valuable lessons from the feedback we received from academia or by the payment industry. The academic opinion was that although it was a good technical solution it presented nothing new, rather the solution combined several existing elements in a novel way. The payment industry opinion was that although interesting, the solution could not be implemented due to the difficulties associated with managing a centralised Public Key Infrastructure for existing the POS terminal network.

In addition; the POS Authentication protocol does not protect against relay attacks and man-in-the-middle attacks as these attacks sit in between an authorised POS terminal and the contactless card. To protect against man-in-the-middle attacks we would add end-to-end encryption into the protocol. This would require the addition step at the beginning of the protocol in which the card and POS terminal perform a symmetric key exchange. Protecting against relay attacks is more problematic. The distance bounding techniques used to prevent relay attacks require both very precise timing of the message time of flight and encryption of the message by the card. The time required by card to encrypt a message is much larger than the time added by the relay to the time of flight, in our practical experiments, Chapter 9, this was approximately 200ms for encryption vs 10ms added by the relay.

From this we concluded that although our strategy was valid to attempt to build a solution which was designed to work within the existing bounds of the EMV protocol, it did not fulfil the academic requirement, nor did it meet the current priorities of the payment industry.

A more productive line of research would be to completely redesign the protocol, rather than creating a compromise solution that fitted into the existing EMV framework. This would allow us to take into consideration the three issues identified in this PhD, vulnerabilities created by the wireless interface, vulnerabilities created by the removal of the PIN in the contactless protocol and vulnerabilities deriving from the complexity of the EMV authentication process.

Chapter 9. Practical Experimental Research

The research presented in this PhD thesis focuses on the EMV protocol as defined by the EMV specifications [6] [7]. The physical instantiation of the EMV specifications are the POS terminals and EMV payments cards we see in our everyday lives. This chapter presents the practical experiments carried out during the course of my PhD research which investigated the reality of the EMV protocol as it is implemented in EMV cards and POS terminals.

- Practical Attack on Contactless Payment Cards. This was a practical experiment presented as a workshop paper at HCI 2011. This experiment was carried out to ascertain if the contactless interface made it easier to skim the card details of an EMV card, *"has usability been improved at the cost of security?"*. This experiment clearly showed this to be the case.
- Contactless Identity Theft. This was a lightning talk paper presented at SOUPS 2013. The theme of the talk was, what are the potential security risks of the increasing number and diversity of contactless cards we are all carrying in our wallet?. It led me to ask two questions "what information can we extract from all of the different ISO 14443 contactless cards in a cardholder's wallet?" and "how does that impact the cardholder's security?".
  Given that a malicious attacker can read all of the contactless cards in your wallet, EMV cards, Oyster cards, ITSO cards etc, what can they find out about us? My practical experiment looked at what data was available from each of the cards in my wallet and could we combine data from several other cards to compromise the EMV payments cards. This turned out to be the case as some ITSO travel cards contained the cardholder's date of birth which creates a problem as it is one of the security questions associated with EMV cards.
- Malicious Multiple Card Reader Software. This software implementation allows us to read all of the contactless cards in a wallet in less than a second. It is not an attack in itself, it is an enabling technology making it possible to exploit vulnerabilities such as the contactless PIN verify, Chapter 7, and the foreign currency flaw, Chapter 6.
- Mobile Phone Attack Platform Demonstrations. Our analysis methodology identifies vulnerabilities in the EMV protocol. With the software emulator we are able to perform exhaustive testing upon the vulnerability to find out the practical impact on EMV cards and POS terminals. These are very much lab based experiments, so to make the impact of the vulnerability visible to the EMV card carrying public, we have developed a number of

Android mobile phone based attack scenarios. The use of an everyday mobile phone clearly demonstrates that this is not just applicable to lab based experiments.

9.1 Practical Attack on Contactless Payment Cards

The intention of this demonstration is to highlight a potential vulnerability in the new contactless payments system and to propose a workable / low cost solution that addresses the issue.

The most disturbing feature of this demonstration Figure 40 is that it is low tech, low cost and relatively easy to implement. The simplicity of the attack is what makes it both accessible and attractive to criminals.

The sophistication required to skim data from EMV Chip & PIN cards is reflected in news reports of card fraud attacks in the UK [80]. Criminals posing as repair engineers in order to replace legitimate Chip & PIN terminals in unsuspecting shops [81] gangs recruiting members of staff in shops and petrol filling stations to replace Chip & PIN terminals [82] gangs building skimming devices which fit over the top of legitimate ATM machines.

This practical experiment shows that I was able to build a working attack scenario which utilised contactless technology to skim the EMV card details and a USB camera to capture the CVV2 from the rear of the card. This is enough information to make online purchase on many websites.

All of the hardware required for this experiment was off-the-shelf costing around £50 and all of the software was downloadable from the internet. In reality this means that the level of sophistication required to commit fraud against contactless cards is significantly lower than that required for EMV Chip & PIN cards.

The specific weakness exploited in this demonstration is that the contactless cards will divulge the (i) card number (ii) cardholder name (iii) expiry date (iv) issue date, unencrypted to any NFC reader that requests the data.

This is not an error, the functionality is fully compliant with the EMV specification for contactless cards. It is assumed that the maximum read range of 10cm makes it difficult to get close enough to read the card without arousing suspicion. This assumption is incorrect.

The demonstration is a mock-up of a counter top, Figure 40, on which a Chip & PIN terminal has been set up to accept legitimate payments from EMV cardholders. It replicates the design of counters found in many UK filling stations and shops, with a display area for sweets immediately adjacent to the POS.

A hidden NFC reader has been set up to capture the details of any contactless card which is inserted into the EMV Chip & PIN terminal. Data capture takes place before card is fully inserted into the terminal so it does not affect the legitimate Chip & PIN transaction.

The CVV2 (Card Verification Value) is the 3 digit security code printed on the back of the card. The purpose of the CVV2 is as a security check value for card-not-present (online and telephone) transactions. Note: not all online retailers require the CVV2.

The demonstration captures the CVV2 using a hidden camera, this technique was selected as it is low cost, easy to implement and it is technology that the criminals are already using in their current attacks.

The aim of the attack is to capture the data required to purchase high value (easy to resell) items online. Many websites will allow the customer to specify a delivery address which is different from the cardholder address registered with the bank (gifts for friends and family), this allows the criminals to receive the stolen goods without raising suspicion.

The £15 transaction limit which is designed to protect contactless cards does not apply as the data is being used to make online purchases.

Figure 40 - Point Of Sale (POS) Mock-up with EMV Terminal

9.1.1 Conclusions from "Practical Attack on Contactless Payment Cards"

This project proved that it was relatively easy to skim data from a contactless card and use that data to purchase goods from an online retailer. I built the demonstration for this project for just £50 and developed the software in 2 weeks with no previous knowledge of the EMV protocol; thereby proving that this type of attack is easily within reach of the talented amateur.

Shortly after this chapter was published Channel 4 News broke the news that Amazon online retailer were not applying the recommended security checks [48] for online payments. Specifically, Amazon do not require the 3 digit CVV2 from the reverse of the card. The CVV2 security check is an extra

precaution designed to ensure that electronic data alone, from the magnetic stripe, the contact chip or the contactless chip, cannot be used to make a *card-not-present* purchase.

9.2 Malicious Multiple Card Reader Software

EMV contactless payment cards are compliant with the ISO 14443 standard for contactless cards. In the UK and in many other countries millions of us now regularly carry several contactless cards in our wallets, the most common being contactless travel cards and building access control cards.

The multiple card reader software utilises the functionality described in ISO-14443 Part 3, Contactless integrated circuit cards: Initialization and anti-collision [83], to identify all of the cards in a wallet and communicate with the cards in turn.

During our research we observed that it was common practice for cardholders to present their entire wallet to a card reader rather than taking the relevant card out of the wallet before presenting it. The malicious multiple card reader software can exploit this behaviour to read all of the cards in the wallet at once. In his book The Art of Deception [84, 23] Kevin Mitnick looks at different strategies for an attacker to socially engineer victims into giving up their security information Mitnick postulates that the best strategy is to *"just ask for it"*. In this case we take advantage of the victim's every day actions in using the door reader, to get the victim to bring his cards to our reader.

The malicious multiple card reader is a flexible attack platform which we have used to demonstrate the impact of vulnerabilities such as, contactless PIN verify (FC 2013) and high value foreign currency attack (CCS 2014).

9.2.1 Multiple Card Reader Implementation

Initialization and anti-collision [83] involves obtaining the Unique Identifiers (UID) of each of the cards in the NFC field. Once this is complete, the UID is used to activate each card individually. The card is then `ready` to accept commands. For successful communication only one card can be `active` at any one time. Table 16 describes the transitions between the different states `idle, ready, active` and `halt` which allow the reader to successfully communicate with an individual card when there are multiple cards in the field.

Table 16 - ISO-14443 Card State Transitions

State	Description
`Idle`	Upon entry to the NFC field all cards will power up into the `idle` state.
`Ready`	The reader transmits `REQA` / `WUPA` command putting the cards into the `ready` state. Once all of the cards are in the `ready` state the anti-collision loop sequence can begin.
`Active`	The anti-collision loop sequence is an iterative process used by the NFC reader to find the UID of the next card in the field. The anti-collision command is repeatedly sent to all cards until only one card answers with a complete UID and no collisions. The UID is then used in the `Select` command which moves that card into the `active` state. At this point the reader can communicate with the card using the card type specific protocol (EMV, MIFARE etc.) or instruct the card to `halt` and store the UID for future use.
`Halt`	To communicate with the next card in the NFC field the reader must `halt` the currently `active` card. Cards can be re-awakened from the `halt` state using the `WUPA`.

The process of communicating with multiple cards is as follows:

- the anti-collision loop finds the UID of each card in turn
- Select(UID) moves the card with the given UID into the `active` state
- the `active` card is now ready for communication with the reader, only one card at a time can be `active`
- `halt` is used to stop communicating with the card and move to the next card

Once anti-collision is complete each card type has its own specific communication protocol. We have implemented protocol sequences for three commonly available card types; EMV payment cards, MIFARE classic door access cards and MIFARE DESFire travel pass cards. Communication with the implemented card types is not affected if an unknown card type is also present in the NFC field, the unknown card type is simply ignored once the anti-collision process has identified its UID. The software utilises hardware commands specific to the NXP PN532 chipset [85] to perform the anti-collision loop, initialisation and card selection.

9.2.2 Results

The test results in this section focus on the time taken to perform each of the steps involved in performing the attack scenarios presented in Chapter 6 and Chapter 7. These results are presented to support our assertion that the delay introduced by the attack would not arouse the suspicions of the users of the door access system.

9.2.2.1 Multiple Card Identification

For the multiple card identification tests we used three of the more popular contactless card types; EMV payment cards, MIFARE classic door access cards and MIFARE DESFire travel pass cards. The test results in Table 17 show the average time (over 60 test runs) to identify each card, when there are multiple cards in the NFC field. Results of the tests show the identification of each card takes longer when more cards are in the field.

Table 17 - Multiple Card Identification Times

Cards in NFC Field	2 cards	3 cards	4 cards	5 cards
Identification of Each Card (ms)	214.36	285.82	305.95	358.30
Standard Deviation (ms)	16.91	16.66	72.54	53.87

The maximum number of cards that the ACR-122U reader (used in our tests) can identify in the NFC field varies by card type. Table 18 shows the maximum number of each card type that the reader could identify and communicate with. The first three rows show tests with a single card type in the NFC field. The following three rows represent wallets containing a mixture of card types, with at least one EMV payment card and one MIFARE classic door access card.

Table 18 - Maximum Cards in NFC field

	EMV payment	MIFARE classic	MIFARE DESFire
Single card type	2 cards	5 cards	4 cards
Multiple card types	2 cards	1 card	
	1 card	1 card	1 card
	1 card	3 cards	

9.2.2.2 Total Attack Time

Table 19 illustrates the total time taken by the verify PIN attack, Chapter 7, on two example wallets *wallet 1* containing one MIFARE classic door access card and one EMV payment card and *wallet 2* containing one MIFARE classic, one EMV and one MIFARE DESFire travel pass. The complete sequence identifies all of the cards present in the wallet and then performs two PIN guesses on the EMV payment card for the contactless verify PIN attack.

Table 19 - Multiple Card Identification and Communication Time

Scenario	Identify Card (ms)	Communication (ms)	Total (ms)
wallet 1	428.73	457.20	855.93
wallet 2	643.09	457.20	1070.29

In summary the test results Table 18 show that it is possible to attack a wallet containing multiple card types. Moreover Table 19 shows that for both *wallet 1* and *wallet 2* the total attack time of around 1 second is fast enough to avoid detection by the cardholder.

9.2.3 Conclusions from "Malicious Multiple Card Reader Software"

In this demonstration we use the ISO14443-Part 3 Anti Collision functionality to identify all of the contactless cards in a wallet and interact with the cards individually. This software provides a powerful platform which can be used to attack any vulnerabilities discovered by our analysis methodology. This could result in a situation where a malicious individual places a multiple card reader on top of a legitimate rapid transport barrier such as the turnstiles at the London underground, giving the attacker access to thousands of card skimmed transactions [29] [42] or thousands of skimmed card details [47] [50] [51].

9.3 Contactless Identity Theft

This research looked at the threats posed by multiple different contactless card types being presented at the same time in a single wallet. The data contained on each individual card is no threat, however, multiple data fields from multiple cards potentially provide the correct combination of data to enable a malicious attack. For instance, EMV contactless cards do not contain the cardholder name, ITSO travel cards contain both the cardholder name and cardholder date of birth. Therefore, the combination of these fields being available at the same time creates a security risk to the cardholder.

We also found that some contactless cards gave out personal information such as; the Japanese Suica contactless travel card which made available the individuals travel history (dates time and locations) and kernel 2 contactless cards which revealed the individuals purchase history in the form of the last 10 transaction on the card.

9.3.1 Conclusion from "Contactless Identity Theft"

The conclusion to this work is that there is a significant risk to the security of an individual when multiple contactless cards are held in a single wallet and the entire wallet is placed on the reader.

9.4 Mobile Phone Attack Platform Demonstrations

We have included in our work a number of practical experiments that allow the general public to visualise the results of our research.

We deliberately created our practical experiments using everyday technology that would be available to the criminals and we also avoided using technology and techniques that would only be accessible in a University lab. This demonstrates to the public and to the payment industry that the work we do is applicable to the real world and not just lab based experiments. Figure 41 shows our practical mobile phone : 1. Card Data Skimming, 2. Transaction Relay, 3. High Value Foreign Currency Flaw, 4. Contactless PIN Verify.

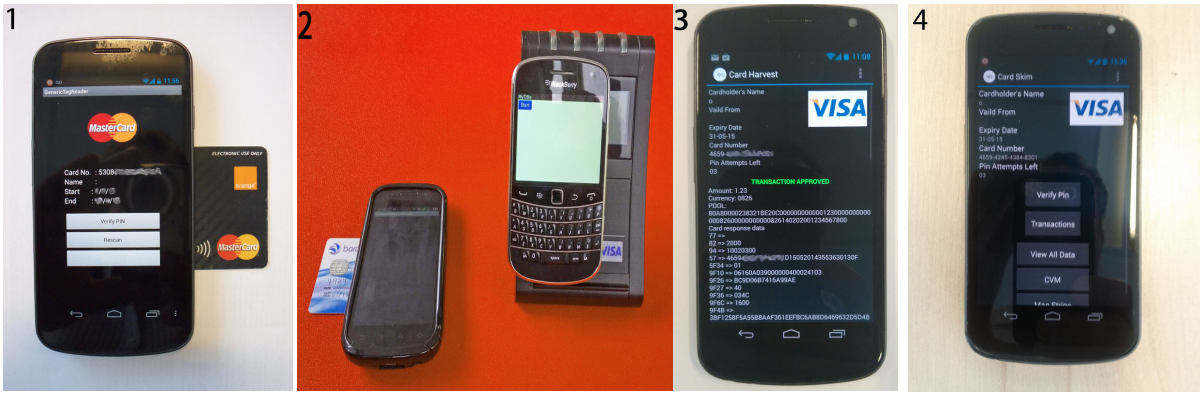


Figure 41 - Practical Experiments for General Public

1. *Card Data Skimming*; this demonstration uses the Android phone's NFC capability to capture the 16 digit card number, expiry date and cardholder name from the contactless card; this is enough information make purchases on Amazon because Amazon do not require the CVV2 security code printed on the back of the card [48]. We were able to show that the we can (i) skim details from a contactless card inside a closed wallet (ii) use those details to create a new Amazon account (iii) buy items from the Amazon store and have them delivered to an address which is different from the address registered for the card. This is a concrete demonstration of the card's vulnerability which members of the public can easily understand.

2. *Transaction Relay;* which uses off-the-shelf mobile phones to relay a contactless transaction from a POS terminal to a contactless card which is still in the victim's wallet. Our practical experiments expanded upon research carried out by Francis et al. (2012) [42]. We implemented our relay over the Internet rather than Bluetooth as used by Francis et al. This extends the attack from one where the attacker must be in the same room into one where the attacker can be in a different country. We used our practical implementation as shown in Figure 41 (2) to carry out timing experiments with the POS terminal in the Computing Science building of Newcastle University campus and the victim card in another building on the campus. The standard transaction time of a kernel 3 contactless transaction is approximately 450milliseconds, relaying the transaction across the University campus increased that time to around 1,100milliseconds which is well within the acceptable contactless transaction time. The fact that our relay works over the Internet would allow an attacker to relay the transaction to another

country, which, when combined with our contactless foreign currency flaw, would allow an attacker to take high value transactions from a victim's card in the UK.

3. *High Value Foreign Currency.* This is the practical demonstration for the research described in Chapter 6. It shows that the attack can be performed using a hardware platform that is readily available, designed to be portable and which would not look suspicious being carried, if challenged by the police; thereby demonstrating the impact of the vulnerability on the millions UK consumers who carry contactless cards.

4. *Contactless PIN Verify.* This is the practical demonstration for the research described in Chapter 7. It is hard to explain in words why the protocol allowing access to the PIN verify command over the contactless interface is detrimental to the security of the card. However, in our demonstration we show an Android phone placed on a closed wallet and correctly guessing the PIN number of the contactless card inside the wallet, which allows people to easily visualise the impact of the flaw.

9.4.1 Conclusions from "Mobile Phone Attack Platform Demonstrations"

When moving our research from the lab to real life scenarios we decided to use mobile phone for the following reasons:

- Our intended audience (the general public) have their own mobile phones and therefore can visualise the impact of the research.
- NFC enabled mobile phones are very accessible
- The Android NFC SDK is comprehensive and easy to use
- The mobile phone is a very portable platform
- Android phones have mobile internet connectivity

9.5 Conclusion

From our practical experiments we learned that it was very easy to attack an EMV contactless card without the cardholder's knowledge. We showed that we could do this with off-the-shelf easy to access hardware and software, demonstrating that the attacks we identified were within the capabilities of malicious individuals. We developed several plausible attack vectors; the hidden reader in the sweetshop, the multiple card reader in the rapid transit system and the Android mobile phone platform.

Unfortunately it is the usability of the wireless interface on EMV contactless cards that makes them vulnerable to these attack vectors whilst the cardholder remains unaware of the attack.

Chapter 10. Conclusion

EMV contactless payments *"has usability been improved at the cost of security?"*. The research presented in this PhD thesis and the literature review show that the functionality that makes contactless payments more convenient (wireless interface and no PIN) can be directly linked to vulnerabilities which have a negative impact on the security of the EMV payments system.

Specifically:

- the wireless interface makes EMV contactless cards vulnerable to new physical attack scenarios whereby the card remains in the cardholder's wallet and the cardholder is unaware of the attack until their bank statement arrives.
- contactless payments do not require a PIN to authorise the transaction, this removes the security step that prevented transactions being made without the explicit consent of the cardholder.
- contactless payments introduce additional complexity to the EMV authentication processes, Figure 9. Our experimental research has shown one specific instance where the difference in the authentication process between the kernel 3 (Visa) and kernel 2 (MasterCard) contactless protocols causes an exploitable vulnerability in kernel 3.

These three factors combine to introduce several new categories of exploitable vulnerability to the EMV payment system, which were not present in Chip & PIN. The new vulnerabilities include, but are not limited to, contactless transaction capture [29] [26] (our primary research interest), contactless skimming [51] [50] [47] contactless eavesdropping [49] [50] contactless transaction relay [42] [46].

At the core of my research is the analysis methodology developed with the assistance of Dr Leo Freitas. We have successfully used the methodology to identify two previously undocumented vulnerabilities in the EMV contactless protocol, being, the foreign currency flaw [29] and contactless PIN verify [27]. We have used practical demonstrations to communicate the impact of our research to the general public and the payments industry. Our collaboration with the payments industry has also helped to increase the impact of research, in that the contactless PIN verify functionality is being removed from UK contactless credit and debit card issued after 2013.

Over the next few years there will be significant security changes in the global card payments system due to the phasing out of magnetic stripe payment cards and the introduction of new payments

technologies, such as mobile phone and wearable device based payments. My future research will continue to use our analysis methodology to examine the new mobile phone based contactless payments technologies, when they are released in the UK. We also need to work on innovative alternatives to the existing transaction protocols in order that they enhance both usability and security together.

10.1 Summary of Contributions

Contributions from my PhD research are:

- an analysis methodology for transaction protocols
- the creation of the protocol emulator
- the identification of two previously undocumented vulnerabilities in the EMV contactless transaction protocol
- a literature review which analyses the security impact of EMV contactless payments in the context of the wider EMV payment system

10.1.1 Analysis Methodology for Contactless Transaction protocols

The analysis methodology developed during my PhD research consists of three elements (i) the protocol emulator which allows us to run practical experiments on the EMV protocol and to inject specific errors into the transaction protocol sequence and test our theories about vulnerabilities (ii) the abstract model of the protocol sequences which highlights any potential security vulnerability in the protocols (iii) UML sequence diagrams and reference tables which precisely document the linkage between our models and the original protocol descriptions in the EMV specifications [6] [7].

The documented linkage to the EMV specifications is the key to our methodology because it demonstrates the difference between a fundamental flaw in the protocol specification and simple implementation errors in the software of EMV cards or POS terminals / ATMs.

The results of the analysis have highlighted a number of significant exploitable flaws (detailed below 10.1.3 and 10.1.4) which allow attackers to subvert the operation of the EMV protocol and thereby compromise the payment system. Exploitable flaws in the EMV payment system fall into a number of different categories our research methodology identifies flaws inherent in the protocol design and ambiguities in the protocol design.

10.1.2 Contribution of the Protocol Emulator

The protocol emulator is fundamental in its contribution and is the cornerstone of my research. It provides a working model of the EMV contactless protocol. It includes both a POS emulation and a card emulation which allows us to model and observe all aspects of the protocol sequence. Its contributions are:

- the protocol emulator code encapsulates our knowledge of the EMV contactless protocol sequences. This is achieved using structured comments embedded in the code which link the running code to the EMV specifications. This creates a knowledge base which can be expanded upon in future research.
- it provides an experimental environment which can execute EMV contactless protocol sequences created to target the vulnerabilities identified by our research. This gives us the ability to demonstrate that these vulnerabilities are exploitable in the real-world.
- the detailed log trace files, generated by the protocol emulator, provide the evidence for the vulnerability being exploitable. They record both the particular parameters that triggered the vulnerability and the UML references which provide a link to its origins in the EMV specifications.

10.1.3 Contactless Foreign Currency Vulnerability

The abstract model highlighted the gap in the specification in EMV specification [6] page 163, which states *"If transaction is in the application currency and is under X value"*, where X is the card offline transaction limit. The specification does not state what to do when the transaction is in a foreign currency. We used the protocol emulator to test UK issued cards, running transactions in GB pounds (the card's native currency) and running in various foreign currencies i.e. US dollars, Euros, Australian dollars.

From these experiments we were able to conclude that it would be possible to build an attack scenario that would allow an attacker with an NFC enabled Android mobile phone to harvest US$999,999.99 transaction approvals from kernel 3 (Visa) cards whilst the card was still in the cardholder's wallet.

Our research also highlighted a fundamental difference between the kernel 3 contactless protocol and the kernel 2 contactless protocols. Kernel 2 has an additional cryptographic request for approval from the card's Issuing bank. This additional check prevents kernel 2 cards from being exploited using the foreign currency vulnerability.

10.1.4 Contactless PIN Verify

Our analysis method highlighted ambiguities in the description of cardholder PIN verification, contained in kernel 3 contactless specification [7] Book C-3. It allows for PIN entry on mobile devices but does not specifically state the correct operation for contactless cards; kernel 3 contactless specifications are ambiguous on the inclusion of the PIN verify from the protocol whereas the kernel 2 specification [68] specifically prohibits PIN verify on contactless cards.

We used the protocol emulator to find which, if any, UK issued kernel 3 cards would allow us to access the Verify PIN functionality over contactless interface. We found that millions of UK issued kernel 3 cards issued were affected by this problem [31]. Further investigation showed that the issue affected cards manufactured with a specific chipset independent of the issuing bank.

10.1.5 Contribution of the Literature Review

The literature review expands our knowledge of the exploitable security vulnerabilities within the EMV payment system. It contributes to the understanding of the common factors which lead to vulnerabilities in the EMV contactless payments. It establishes a link between the existing research and our research themes:

- exploitation of the wireless interface to gain unauthorised access to the contactless card
- no cardholder verification required, this allows unauthorised transactions to be processed without cardholder interaction.
- the complexity of the authentication process contributes to make vulnerabilities exploitable.

10.2 Future Work

The EMV payments system continues to evolve as new payments technologies are introduced and legacy technologies are retired. We are currently in a period of rapid change with new payments technologies being introduced every few years rather than decades.

10.2.1 Continuing Protocol Analysis Research

Analysis of mobile phone payments technologies, such as Google Wallet, will present many interesting new challenges as many of these technologies are based on the magnetic stripe contactless transaction protocol currently prevalent in the USA. Research by Roland and Langer (2013) shows that there are vulnerabilities in the magnetic stripe contactless protocol, it also shows that EMV contactless cards are affected by this vulnerability as the EMV specification requires them to be compatible with magnetic stripe. Future research should therefore look at the magnetic stripe contactless protocol and its impact on the EMV contactless protocol.

10.2.2 New Payments Technologies

There are a number of emerging NFC enabled smart device payment technologies (e.g. mobile phone wallets, rechargeable festival wristbands and smart watches), these technologies utilise the EMV contactless transaction protocol. There are also new technologies for receiving payments using your mobile phone such as iZettle and PayPal POS terminal. These terminals make it much easier for a small trader to take payments on the move, unfortunately they also make it easier for fraudsters to create money mule accounts to receive fraudulent payments. Future research should look at the potential security implications of these new POS terminal technologies.

10.2.3 Continuous Authentication

Future research will look at the benefits, drawbacks and the accuracy of biometric continuous authentication technologies, such as MasterCard payments wristband which continuously monitors the wearer's heart beat ECG [16] [17]. Can continuous authentication be integrated into a more secure and more useable transaction protocol?

BIBLIOGRAPHY

[1] US Federal Reserve, "Change Is Coming: What the EMV Migration May Mean," US Federal Reserve, April 2015. [Online]. Available: http://www.kc.frb.org/publicat/psr/briefings/psr-briefingapr2015.pdf. [Accessed 04 May 2015].

[2] US Federal Reserve, "Payments Study 2013," US Federal Reserve, 2013.

[3] Visa, "Visa Announces US Participation in Global POS Counterfeit Liability Shift," Visa, 2011.

[4] International Standards Organisation, "ISO 7813: Information technology. Identification cards. Financial transaction cards," 2006.

[5] EMVCo, "Worldwide EMV Card and Terminal Deployment," January 2014. [Online]. Available: http://www.emvco.com/about_emvco.aspx?id=202. [Accessed 03 June 2014].

[6] EMVCo, "EMV Specifications for Payment Systems (version 4.3)," November 2011. [Online]. Available: http://www.emvco.com/specifications.aspx?id=223. [Accessed 22 August 2014].

[7] EMVCo, "EMV Contactless Specifications for Payment Systems – Version 2.4," 28 Februrary 2014. [Online]. Available: http://www.emvco.com/specifications.aspx?id=21. [Accessed 10 March 2016].

[8] International Standards Organisation, "ISO 7811-1 Identification cards. Recording technique. Embossing," 2014.

[9] Sophos, "Chip & PIN compatibility leads to insecurity," 22 03 2011. [Online]. Available: https://nakedsecurity.sophos.com/2011/03/22/chip-and-pin-compatibility-leads-to-insecurity/. [Accessed 07 03 2015].

[10] APACS, "Card Fraud the Facts 2005," UK Cards Association, London, 2005.

[11] Financial Fraud ActionUK, "Fraud The Facts 2014," 2014.

[12] Smart Card Alliance, "EMV: Facts at a Glance," 23 06 2011. [Online]. Available: http://www.smartcardalliance.org/resources/pdf/EMV_Facts_20110623.pdf. [Accessed 18 12 2015].

[13] Murdoch SJ, Drimer S, Anderson R and Bond M, "Chip & PIN is Broken," in *IEEE Symposium on Security and Privacy*, Oakland, CA, 2010.

[14] “KrebsonSecurity,” 24 11 2014. [Online]. Available: http://krebsonsecurity.com/. [Accessed 24 11 2014].

[15] Darwin C., On the Origin of Species (by Means of Natural Selection), John Murray, 1859.

[16] Nymi, “Nymi White Paper,” 19 November 2013. [Online]. Available: https://www.nymi.com/news/whitepaper/.

[17] Reuters, “MasterCard, RBC to test if the heart is always true, for payments at least,” 03 11 2014. [Online]. Available: http://www.reuters.com/article/2014/11/03/mastercard-rbc-bionym-idUSL1N0ST11K20141103. [Accessed 29 03 2015].

[18] A. S. H. C. O. C. Tanviruzzaman M, “ePet: When Cellular Phone Learns to Recognize Its Owner,” in *SafeConfig’09,*, Chicago, 2009.

[19] Isaac M and Chen B,, “Apple Pay Gives Glimpse of Mainstream Appeal for Mobile Payments,” *New York Times, The,* 14 11 2014.

[20] CNET, “NFC mobile payments disappoint while money transfers boom,” 04 06 2013. [Online]. Available: http://www.cnet.com/uk/news/nfc-mobile-payments-disappoint-while-money-transfers-boom/. [Accessed 11 03 2015].

[21] Barclaycard, “Barclaycard launches the bPay band: a wristband to pay your way,” Barclaycard, 11 June 2014. [Online]. Available: http://www.barclaycard.com/news/bpay-band-launches.html. [Accessed 10 May 2015].

[22] MasterCard, “Security Matters - Insights on Advancing Security and Fraud Management for Payment Cards,” 2012. [Online]. Available: https://www.mastercard.com/us/wce/PDF/PSI_Magazine_SecurityMatters_US.pdf. [Accessed 13 January 2016].

[23] Murdoch SJ., “Defending against wedge attacks in Chip & PIN,” 25 08 2009. [Online]. Available: https://www.lightbluetouchpaper.org/2009/08/25/defending-against-wedge-attacks/. [Accessed 14 02 2014].

[24] Degabriele, J.P., Lehmann, A., Paterson, K.G., Smart, N.P., Strefler, M, “On the Joint Security of Encryption and Signature in EMV,” in *The Cryptographers’ Track at the RSA Conference 2012.*, 2012.

[25] Bond M, Choudary O, Murdoch S.J, Skorobogatov S, Anderson R, “Chip and Skim: cloning EMV cards with the pre-play attack,” in *35th IEEE Symposium on Security and Privacy*, 2014.

[26] Roland M. and Langer J., "Cloning credit cards: a combined pre-play and downgrade attack on EMV contactless," in *7th USENIX conference on Offensive Technologies*, Berkeley, 2013.

[27] Emms M, Arief B, Little NJ, Van Moorsel A., "Risks of Offline Verify PIN on Contactless Cards.," in *17th International Conference, Financial Cryptography and Data Security.*, 2013.

[28] De Ruiter, J. and Poll, E., "Formal analysis of the EMV protocol suite," in *2011 International conference on Theory of Security and Applications (TOSCA'11)*, Saarbrucken, Germany, 2011.

[29] Emms M, Arief B, Freitas L, Hannon J, and Van Moorsel A, "Harvesting High Value Foreign Currency Transactions from EMV Contactless Credit Cards without the PIN," in *21st Conference on Computer and Communications Security (CCS 2014)*, Arizona, 2014.

[30] Barisani A, Bianco D, Laurie A, Franken Z, "Chip & PIN is definitely broken," in *DEFCON 19*, Las Vagas, NV, 2011.

[31] Barclays, "Heading into a Contactless World," 03 April 2014. [Online]. Available: http://www.newsroom.barclays.com/r/2879/heading_into_a_contactless_world. [Accessed 10 March 2016].

[32] Ouerdi, N.; Ziane, M.; Azizi, A.; Azizi, M.; Lanet, J, "Abstract tests based on SysML models for EMV Card," in *2013 National Security Days (JNS3)*, 2013.

[33] De Koning Gans G. and De Ruiter J., "The smartlogic tool: Analysing and testing smart card protocols," in *2012 IEEE Fifth International Conference on Verification and Validation (ICST)*, 2012.

[34] Choudary O., *The Smart Card Detective: a hand-held EMV interceptor.,* Cambridge Univesity, 2010.

[35] Aarts F, de Ruiter J, and Poll E, "Formal Models of Bank Cards for Free," in *6th International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2013.

[36] Pasquet M., Reynaud J. and Rosenberger C., "Secure Payment with NFC Mobile Phone in the SmartTouch Project," in *International Symposium on Collaborative Technologies and Systems*, 2008.

[37] Le Parisien, "The unstoppable credit card scam," Le Parisien, 24 January 2012. [Online]. Available: http://www.leparisien.fr/faits-divers/l-imparable-escroquerie-a-la-carte-bancaire-24-01-2012-1826971.php. [Accessed 25 April 2015].

[38] Anderson R, Bond M and Murdoch S, "Chip & SPIN," *Computer Security Journal,* vol. 22, no. 2, pp. 1-6, 2006.

[39] MasterCard, "PayPass M/Chip Acquirer Implementation Requirements (version 1.0)," MsaterCard, 2008.

[40] International Standards Organisation, "ISO 14443 Contactless Integrated Circuit Cards," 2011.

[41] Francis L, Hancke G, Mayes K, Markantonakis K., "Potential Misuse of NFC Enabled Mobile Phones with Embedded Security Elements as Contactless Attack Platforms.," in *International Conference for Internet Technology and Secured Transactions (2009)*, 2009.

[42] Francis L, Hancke G, Mayes K, Markantonakis K., "Practical Relay Attack on Contactless Transactions by Using NFC Mobile Phones.," in *The 2012 Workshop on RFID and IoT Security (RFIDsec 2012 Asia)*, 2012.

[43] Drimer, S. and Murdoch, S.J., "Keep Your Enemies Close: Distance Bounding Against Smartcard Relay Attacks," in *16th USENIX Security Symposium*, Boston, MA, USA, 2007.

[44] Kfir Z. and Wool A., "Picking Virtual Pockets using Relay Attacks on Contactless Smartcard Systems," in *First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, 2005.

[45] Roland M., Scharinger J., "Applying Relay Attacks to Google Wallet," in *5th International Workshop on Near Field Communication*, Zurich, 2013.

[46] Roland M, Langer J, and Scharinger J, "Relay Attacks on Secure Element-enabled Mobile Devices: Virtual Pickpocketing Revisited," in *27th IFIP TC 11 Information Security and Privacy Conference, SEC 2012*, Heraklion, Crete, Greece,, 2012.

[47] Sky News, "Contactless Cards: App Reveals Security Risk," Sky News, 04 June 2013. [Online]. Available: http://news.sky.com/story/1099259/contactless-cards-app-reveals-security-risk. [Accessed 29 April 2015].

[48] Channel 4 News, "Millions of Barclays card users exposed to fraud," Channel 4 News, 23 March 2012. [Online]. Available: http://www.channel4.com/news/millions-of-barclays-card-users-exposed-to-fraud. [Accessed 28 August 2014].

[49] Diakos T.P., Briffa J.A., Brown T.W.C.and Wesemeyer S., "Eavesdropping near-field contactless payments: a quantitative analysis," *The Journal of Engineering,* no. September 2013, 2013.

[50] G. Hancke, “Practical Eavesdropping and Skimming Attacks on High-Frequency RFID Tokens.,” *Journal of Computer Security,* vol. Vol 19, no. Issue 2, pp. pp. 259-288, 2011.

[51] Emms M., “Practical Attack on Contactless Payment Card,” in *HCI2011 Workshop - Heath, Wealth and Identity Theft (2011)*, 2011.

[52] Kirshenbaum I. and Wool A., “How to Build a Low-Cost, Extended-Range RFID Skimmer,” in *15th USENIX Security Symposium*, 2006.

[53] Oren Y, Schirman D, Wool A, “Range Extension Attacks on Contactless Smart Cards,” in *ESORICS 2013*, 2013.

[54] Markantonakis K, Tunstall M, Hancke G, Askoxylakis I and Mayes K, “Attacking smart card systems: Theory and practice,” *Science Direct,* vol. 14, no. 2, 2009.

[55] Anderson R., “RFID and the middleman,” in *11th International Conference on Financial cryptography*, 2007.

[56] Google, “Android API Guides - Near Field Communication,” 2014. [Online]. Available: https://developer.android.com/guide/topics/connectivity/nfc/index.html. [Accessed 24 May 2015].

[57] Woodcock J, Freitas L, “Z/Eves and the Mondex Electronic Purse,” in *Third International Colloquium, Theoretical Aspects of Computing - ICTAC 2006*, Tunis, 2006.

[58] Woodcock, J. and Davies, J, Using Z., Prentice Hall, 1998.

[59] Jones C and Woodcock J., “The certification of the Mondex electronic purse to ITSEC Level E6,” *Formal Aspects of Computing,,* vol. 20, no. 1, January 2008.

[60] Freitas L, Woodcock J., “Mechanising Mondex with Z/Eves,” *Formal Aspects of Computing,* vol. 20, no. 1, pp. 117-139, 2008.

[61] Freitas L, and Emms M., “Formal specification of EMV protocol. (TR 1429),” Newcastle University - School of Computing Science Technical Report Series, Newcastle, 2014.

[62] Cooper, D. and Barner, J, “Tokeneer ID station EAL5 demonstrator,” Altran Praxis, 2008.

[63] Smans, J., Jacobs, B., and Piessens, “VeriFast for Java: A Tutorial. In,” *Aliasing in Object-Oriented Programming,* vol. vol. 7850, p. pp. 407– 442, 2013.

[64] J. Reason, "The Contribution of Latent Human Failures to the Breakdown of Complex Systems," *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences,* vol. 327, no. 1241, pp. 475-484, 1990.

[65] UK Cards Association Limited, "Standard 70 – Card Acceptor to Acquirer Interface Standards," 2013.

[66] International Organization for Standardization, "ISO 8583:1995 – Financial transaction card originated messages –Interchange message specifications," 1995.

[67] MasterCard, "PayPass – M/Chip Technical Specifications (version 1.3)," 2005.

[68] MasterCard, "M/Chip, Acquirer Implementation Requirements," 2006.

[69] Bonneau J. Preibusch S., Anderson R, "A birthday present every eleven wallets? The security of customer-chosen banking PINs.," 2012.

[70] Choudary O., "The Smart Card Detective: a hand-held EMV interceptor.," in *Cambridge*, 2010.

[71] G. Willey, *PIN Number burglar used victims' card,* 2012.

[72] Advanced Card Systems, "ACR122U NFC Reader Application Programming Interfac," 2011.

[73] Oracle, *Oracle: Java Smart Card I/O API,* 2012.

[74] National Security Agency, "The Case for Elliptic Curve Cryptography (2009)," 2009. [Online]. Available: http://www.nsa.gov/business/programs/elliptic_curve.shtm.

[75] EMVCo, "Book 2 Security and Key Management Version 4.1z ECC With support for Elliptic Curve Cryptography (2007)," 2007.

[76] Brown, D.R.L, Campagna, M.J., Vanstone, S.A, "Security of ECQV-Certified ECDSA Against Passive Adversaries.," in *IACR Cryptology ePrint Archive*, 2009.

[77] Brown, D.R.L., Gallant, R.P., Vanstone, S.A, "Provably secure implicit certificate schemes.," in *5th International Conference, Financial Cryptography and Data Security*, 2001.

[78] Sun Microsystems, "Java Card Platform 2.2.2 - Application Programming Interface," 2006.

[79] Antipa, A., Brown, D.R.L., Gallant, R.P., Lambert, R.J., Struik, R., Vanstone, S.A, "Accelerated Verification of ECDSA Signatures.," in *12th International Workshop, Selected Ar-eas in Cryptography.*, 2005.

[80] BBC News, “Threat to Chip & PIN Terminals,” 2010. [Online]. Available: http://news.bbc.co.uk/1/hi/business/8437606.stm.

[81] BBC News, “Device Steals Chip & PIN Data,” 2008. [Online]. Available: http://news.bbc.co.uk/1/hi/business/7557956.stm.

[82] BBC News, “ATM Fraud Gang Jailed,” 2008. [Online]. Available: http://news.bbc.co.uk/1/hi/england/7798446.stm.

[83] ISO, “ISO 14443-3 Contactless Integrated Circuit Cards - Initialization and anticollision,” 2011.

[84] Mitnick K., The Art of Deception, Indianapolis, Indiana; United States of America: Wiley Publishing Ltd, 2002.

[85] NXP, “PN532 User Manual,” NXP, 2007.

[86] [Online].

Appendix A. Analysis Methodology Example

This appendix illustrates the how the elements of our analysis methodology are linked together. It takes as the example of the Kernel 3 fDDA contactless transaction protocol sequence (kernel 3). The example focuses on a single command in protocol sequence showing its representation in the UML diagram, the reference table, the protocol emulator Java code and the Z abstract model.

A.1 UML Diagram – Kernel 3 Kernel 3 fDDA Contactless Transaction

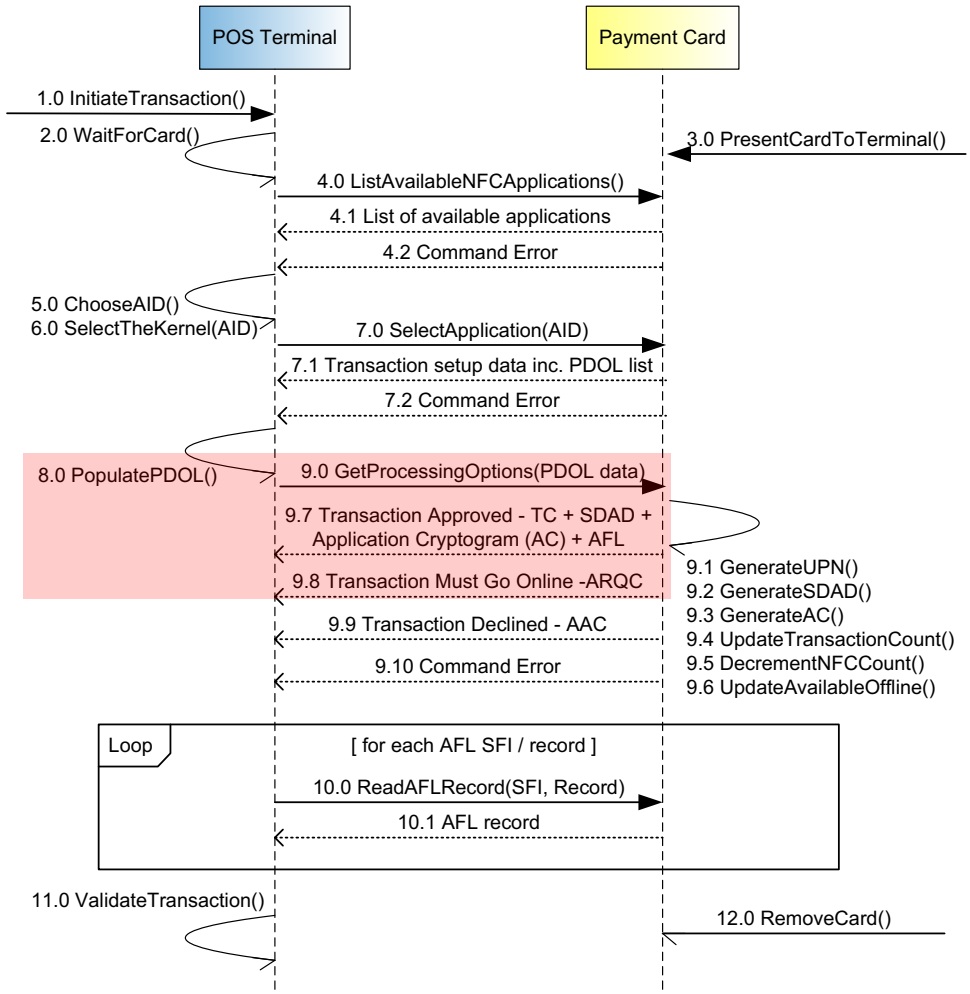


Figure 42 – UML Diagram - Kernel 3 fDDA Contactless Transaction

A.2 Reference Table – Kernel 3 fDDA Contactless Transaction

Table 20 - Reference Table – Kernel 3 fDDA Contactless Transaction

Descriptive Text	References
➔ 1.0 InitiateTransaction() The terminal initiates the transaction by setting the required transaction data, this will typically include: • transaction amount • transaction date • transaction currency • country code for the merchant / transaction • unpredictable number generated by the POS terminal, • the terminal qualifiers which specify the capabilities of the POS terminal The Terminal Transaction Qualifiers (TTQ) is a data structure is a list of the capabilities supported by the POS terminal (i.e. online, offline, Magnetic Stripe, EMV, Contactless. PIN authorisation, signature authorisation).	EMV v2.2 Book C-3 - p8 2.1 EMV Mode Configuration EMV v2.2 Book C-3 - p106 A.2 Data Elements by Name EMV v2.2 Book C-3 - p127 Annex C Fast Dynamic Data Authentication EMV v2.2 Book A - p23 5.6.5 Kernel and Entry Point Configuration Data EMV v2.2 Book A - p25 5.7 Transaction Data EMV v4.3 Book 3 - p91 10.1 Initiate Application Processing
➔ 2.0 WaitForCard() The POS terminal powers up its NFC reader and waits for a contactless card to come into the NFC field. The reader software performs the ISO-14443:Part 3 Initialization and anti-collision sequence. The device driver then notifies the host software of the UID and historical bytes from the card. This process is handled in the device driver of the NFC reader.	ISO/IEC 14443-3:2011 – p5 6 Type A Initialization and anti-collision ISO/IEC 14443-4:2008 – p4 5 Protocol Activation of PICC Type A
➔ 3.0 PresentCardToTerminal() When the card enters the NFC field it is powered up and supplies it's UID to the reader in response to the ISO-14443:Part 3 Initialization and anti-collision sequence. The card is now active and ready to communicate with the terminal.	ISO/IEC 14443-3:2011 p5 6 Type A Initialization and anti-collision ISO/IEC 14443-4:2008 – p4 5 Protocol Activation of PICC Type A
➔ 4.0 ListAvailableNFCApplications() Terminal will use the Proximity Payment System Environment (PPSE) command to list the valid payment application on the card. The terminal sends the SELECT("2PAY.SYS.DDF01") command to the card.	EMV v2.2 Book B – p16 3.3 Combination Selection EMV v2.2 Book B – p17 3.3.1 PPSE Data for Application Selection p17 EMV v4.3 Book 1 – p127 11.3 SELECT Command-Response APDUs EMV v4.3 Book 1 - p133 12. Application Selection EMV v4.3 Book 1 - p135 12.2.2 Structure of the PPSE
➔4.1 List of available applications EMV compliant contactless payment card will respond with a list of all of the available applications (AIDs) on the card. The list also includes the associated priority values (which define the order which the AID should be used by the terminal) and the Kernel ID (which for contactless transactions defines the protocol sequence required).	EMV v2.2 Book B - p18 Table 3-2: SELECT Response Message Data Field (FCI) of the PPSE EMV v2.2 Book B - p24 3.3 Combination Selection (3.3.25) EMV v2.2 Book B - p19 Table 3-3: Format Application Priority Indicator EMV v2.2 Book B - p20

	Table 3-4: Format Kernel Identifier EMV v4.3 Book 1 - p127 11.3 SELECT Command-Response APDUs EMV v4.3 Book 1 - p144 12.3 Building the Candidate List IS0 7816-4 - p65 8.2.1.2 Application identifier -
←4.2 Command Error For all cases where the card does not support the PPSE command it will return a failure status code[1], in this case the outcome of the transaction is set to "End the Application" and the POS terminal exits the transaction. [1] The success status code is 0x9000 all other status codes are failure codes.	EMV v2.2 Book B - p23 3.3 Combination Selection (Step 1) EMV v2.2 Book B - p28 3.3 Combination Selection (Step 3) EMV v2.2 Book B - p33 3.5 Outcome Processing (3.5.1.5 Other)
↘ 5.0 ChooseAID The terminal selects the contactless Kernel and Application Identifier (AID) to be used for the transaction. The application selected should be the highest priority application supported by both the card and the POS terminal. The applications supported by the card is returned by "4.1 List of available applications", the list contains Application Identifier (AID), the Application Priority Indicator, the Kernel ID and the name of the application.	EMV v4.3 Book 1 - p144 12.3 Building the Candidate List EMV v2.2 Book B - p16 3.3 Combination Selection EMV v2.2 Book B - p19 Table 3-2: Format Application Priority Indicator EMV v2.2 Book B - p20 Table 3-4: Format of the Kernel Identifier EMV v2.2 Book B - p31 3.4 Kernel Activation EMV v2.2 Book A - p30 5.8.2 Application Selection & Kernel Activation
↘ 6.0 SelectKernel(AID) The kernel is selected based on Kernel ID ("03" Visa, "02" MasterCard, "04" Amex or "05" JCB) returned by the card in "4.1 List of available applications". If the Kernel ID is not included in the list then the POS terminal will select the kernel based on the Registered Application Provider Identifier (RID) contained in the first five digits of the AID.	
→7.0 SelectApplication(AID) The fDDA transaction sequence is specific to Visa cards for which the usual AID will be A0000000031010, the command would therefore be SELECT(A0000000031010) to select the Visa application.	EMV v2.2 Book B - p29 3.3.3 Final Combination Selection EMV v2.2 Book B - p31 3.4 Kernel Activation EMV v2.2 Book B – p16 3.3 Combination Selection p16 EMV v4.3 Book 1 - p127 11.3 SELECT Command-Response APDUs
←7.1 Transaction Setup Data including PDOL list If the Visa application is successfully selected the card will return the data that the terminal requires to set up the transaction including the PDOL list. The Processing Data Objects List (PDOL) is a list of data elements the card requires to complete the transaction, the terminal returns the populated PDOL data in the Get Processing Options command. Typically the data fields requested by the card wuill include	EMV v2.2 Book C-3 – p12 2.4.1 Initiate Application Processing EMV v4.3 Book 3 - p91 10.1 Initiate Application Processing EMV v2.2 Book B - p33 3.5 Outcome Processing (3.5.1.5 Other) EMV v4.3 Book 4 - p115 Annex A - Coding of Terminal Data Elements

the transaction amount, currency, date, country and TTQ	
←7.2 Command Error SelectApplication(AID) will fail if the application AID was not on the list of available list of available applications (4.1) or the application AID is not valid for contactless operation.	EMV v2.2 Book B - p23 3.3 Combination Selection (Step 1) EMV v2.2 Book B - p28 3.3 Combination Selection (Step 3) EMV v2.2 Book B - p33 3.5 Outcome Processing (3.5.1.5 Other)
↘ 8.0 PopulatePDOL() The Processing Options Data Objects List (PDOL) is the list of data elements that the card requires to approve the transaction. The terminal populates the PDOL with data, the data elements would usually include transaction amount, currency, unpredictable number, transaction date.	EMV v2.2 Book B - p34 3.6 Data Element Processing EMV v2.2 Book C-3 - p12 2.4.1 Initiate Application Processing EMV v2.2 Book C-3 - p40 5.1 Initiate Application Processing EMV v4.3 Book 3 – p38 5.4 Rules for Using a Data Object List (DOL) EMV v4.3 Book 4 - p115 Annex A - Coding of Terminal Data Elements EMV v4.3 Book 3 – p163 Annex C3 Cardholder Verification Rule Format EMV v2.2 Book C-3 - p110 Annex A.2 Data Elements by Name (9F66) EMVv2.2 Book C-3 - p127 Annex C Fast Dynamic Data Authentication
→9.0 GetProcessingOptions(PDOL data) In the Kernel 3 fDDA transaction Get Processing Options (GPO) is used to request completion of the transaction. The PDOL data must contain all of the data elements requested by the card otherwise the transaction will be rejected.	EMV v2.2 Book C-3 - p12 2.4.1 Initiate Application Processing EMV v2.2 Book C-3 - p40 5.2 Initiate Application Processing EMV v2.2 Book C-3 - p40 5.2.1 Get Processing Options (GPO) Command EMV v2.2 Book C-3 - p40 to p46 5.2.2 Initiate Application Processing EMV v4.3 Book 3 - p60 6.5.8.4 Data Field Returned in the Response
↘ 9.1 GenerateUPN() The card generates a 4 byte unpredictable number (UPN) *?? 8-byte GenerateAC ??*. The card UPN is incorporated into the signed transaction data SDAD and Application Cryptogram (AC). This injects random data into the SDAD signature and the MAC of the AC thereby protecting them from partial oracle attacks on the cryptography.	EMV v2.2 Book C-3 - p89 A.2 Data Elements by name EMV v2.2 Book C-3 - p109 A.2 Data Elements by name EMV v2.2 Book C-3 - p127 Annex C Fast Dynamic Data Authentication
↘ 9.2 GenerateSDAD() ??? MJE to Complete ???	
↘ 9.3 GenerateAC()	
↘ 9.4 UpdateTransactionCount()	
↘ 9.5 DecrementNFCCount()	
↘ 9.6 UpdateAvailableOffline()	

←9.7 Transaction Approved - TC, SDAD, Application Cryptogram (AC), AFL If the card approves the completion of the transaction in offline mode, it will return Transaction Cryptogram (TC) in the Cryptogram Information Data (CID). The card also returns all of the data elements required by the terminal to complete the transaction: Signed Dynamic Application Data (SDAD) used by the terminal to verify that the card has approved the same transaction that the terminal sent. Application Cryptogram (AC) used in the completion of the transaction with the Bank to validate that a valid card completed the transaction. Application File Locator (AFL) contains the location in the card's file structure where the terminal can read the data elements required to complete the transaction.	EMV v2.2 Book C-3 - p50 5.4.3 Determine the Card Disposition EMV v2.2 Book C-3 - p43 5.2.2.2 GPO Response SW1 SW2 EMV v2.2 Book C-3 - p46 5.2.2.3 Contactless Path Determination EMV v2.2 Book C-3 - p97 A.2 Data Elements by Name EMVv2.2 Book C-3 - p127 Annex C Fast Dynamic Data Authentication EMV v2.2 Book C-3 - p128 C.1 Dynamic Signature Verification EMV v4.3 Book 3 - p59 6.5.8 Get Processing Options APDUs EMV v4.3 Book 3 - p60 6.5.8.4 Data Field Returned in the Response
← 9.8 Transaction Must Go Online –ARQC If the card requires online completion of the transaction it will return ARQC in the Cryptogram Information Data (CID). Online completion is required when the amount of the transaction exceeds the cards offline transaction limit or offline cumulative limit or when the number of offline transactions exceeds the number of consecutive offline transactions.	EMV v2.2 Book C-3 - p50 5.4.3 Determine the Card Disposition EMV v2.2 Book C-3 - p97 A.2 Data Elements by Name EMV v2.2 Book C-3 - p14 2.4.7 Online Processing (EMV Mode) EMV v2.2 Book C-3 - p15 2.4.8 Completion (EMV Mode)
← 9.9 Transaction Declined - AAC If the card declines the transaction it returns AAC in the CID. The card declines the transaction when it cannot be completed as requested by the terminal.	EMV v2.2 Book C-3 - p50 5.4.3 Determine the Card Disposition EMV v2.2 Book C-3 - p97 A.2 Data Elements by Name EMV v2.2 Book C-3 - p14 2.4.7 Online Processing (EMV Mode) EMV v2.2 Book C-3 - p15 2.4.8 Completion (EMV Mode)
← 9.10 Command Error The possible error codes returned by Get Processing Options are: 6984- Try Another Interface. The transaction should be reattempted using either the contact interface or the magnetic stripe interface. 6985 - Select Next. The transaction should be reattempted using the next combination of Kernel / AID (if any). 6986 - Try Again, reattempt the transaction with the same parameters. The card must be represented, this may occur if the card is removed too early.	EMV v2.2 Book C-3 - p43 5.2.2.2 GPO Response SW1 SW2 EMV v2.2 Book C-3 - p15 2.4.8 Completion (EMV Mode) EMV v2.2 Book B - p24 3.3 Combination Selection
Loop [for each AFL SFI / record] →10.0 ReadAFLRecord(SFI, Record) The Application File Locator (AFL) returned in the GPO response contains the location of the data records which contain the data elements that the terminal will require to complete the transaction. The AFL contains a range of SFI / Records. The data the AFL points to will include the public keys required to decrypt the SDAD returned in the GPO response.	EMV v2.2 Book C-3 - p47 5.3 Read Application Data EMV v4.3 Book 3 - p93 10.2 Read Application Data EMV v4.3 Book 3 - p65 6.5.11 Read Record Command-Response APDUs

←10.1 AFL record The card will return the requested SFI / Record.	EMV v4.3 Book 3 - p93 10.2 Read Application Data EMV v4.3 Book 3 - p65 6.5.11 Read Record Cmd-Response APDUs
→11.0 ValidateTransaction() If the card has approved the transaction Offline Data Authentication is performed by the terminal to verify the dynamic signature and authenticate the data returned by the card in response to the GPO command. The terminal verifies the DDA dynamic signature contained in the SDAD using the CA public key, issuer public key and the card's public key. The resulting data packet is a SHA-1 hash of the terminal unpredictable number, the card unpredictable number, the transaction amount, the currency and the date. This process ensures that the card has verified the transaction without altering any of the values in the transaction. The unpredictable numbers are used to ensure that transactions cannot be recorded and replayed. If the signed data in the SDAD passes the validation the transaction is deemed valid and the terminal will approve the transaction.	EMV v2.2 Book C-3 - p58 5.6 Offline Data Authentication EMV v4.3 Book 2 - p51 6 Offline Dynamic Data Authentication EMV v4.3 Book 2- p60 6.2 Retrieval Certification Authority Public Key EMV v4.3 Book 2- p60 6.3 Retrieval of Issuer Public Key EMV v4.3 Book 2- p63 6.3 Retrieval of ICC Public Key EMV v4.3 Book 2- p66 6.5 Dynamic Data Authentication (DDA) EMV v4.3 Book 2- p68 6.5.2 Dynamic Signature Verification
←12.0 RemoveCard() When the last record listed in the AFL has been successfully returned, the terminal will indicate to the cardholder that the card can be removed from the reader.	EMV v4.3 Book 3 - p48 5.4 Card Read Complete

A.3 Protocol Emulator Code – Kernel 3 fDDA Contactless Transaction

Table 21 – Protocol Emulator Code - Kernel 3 fDDA Contactless Transaction

```
public class Kernel3 extends Kernel
{

/**
 * Kernel 3 - Visa EMV Mode Contactless Transactions (fDDA)
 * UML diagram – Kernel 3 (Visa) fDDA Protocol Sequence
 * This Method performs steps 8.0 PopulatePDOL() to 11.0 ValidateTrasnaction().
.* This method starts once the payment application has been selected,
 * steps 1.0 InitiateTransaction() to 7.0 SelectApplication() are performed by
.* the common Kernel processes.
 * Concrete implementation of abstract methods for EMV mode transactions.
 * UK issued cards should operate in EMV mode and US issued cards should
 * operate in Magnetic Stripe Mode
 * References
 * EMV v2.2 Book B - 3.6 Data Element Processing - p34
 * EMV v2.2 Book C-3 - 2.4.1 Initiate Application Processing - p12
 * EMV v2.2 Book C-3 - 5.1 Initiate Application Processing - p40
 * EMV v4.3 Book 3 - 5.4 Rules for Using a Data Object List (DOL)
 * EMV v4.3 Book 4 - Annex A - Coding of Terminal Data Elements - p115
 * EMV v2.2 Book C-3 - A.2 Data Elements by Name (9F66) - p110
 * EMV v2.2 Book C-3 - Annex C Fast Dynamic Data Authentication - p127
 * EMV v2.2 Book C-3 - 5.2.1 Get Processing Options (GPO) Command - p40
 * EMV v2.2 Book C-3 - 5.2.2 Initiate Application Processing - p40 to p46
 * EMV v4.3 Book 3 - 6.5.8.4 Data Field Returned in the Response - p60
 * EMV v2.2 Book C-3 - 5.4.3 Determine the Card Disposition - p50
 * EMV v2.2 Book C-3 - 5.2.2.2 GPO Response SW1 SW2 - p43
 * EMV v2.2 Book C-3 - 5.2.2.3 Contactless Path Determination - p46
 * EMV v2.2 Book C-3 - A.2 Data Elements by Name - p97
 * EMV v2.2 Book C-3 - C.1 Dynamic Signature Verification - p128
 * EMV v4.3 Book 3 - 6.5.8 Get Processing Options APDUs - p59
 * EMV v2.2 Book C-3 - 5.3 Read Application Data - p47
 * EMV v4.3 Book 3 - 10.2 Read Application Data - p93
 * EMV v4.3 Book 3 - 6.5.11 Read Record Command-Response APDUs - p65
 * EMV v2.2 Book C-3 - 2.4.2 Read Application Data p13
 * EMV v2.2 Book C-3 - 5.6 Offline Data Authentication - p58
 * EMV v4.3 Book 2 - 6 Offline Dynamic Data Authentication - p51
 * EMV v4.3 Book 2- 6.2 Retrieval Certification Authority Public Key - p60
 * EMV v4.3 Book 2- 6.3 Retrieval of Issuer Public Key - p60
 * EMV v4.3 Book 2- 6.3 Retrieval of ICC Public Key - p63
 * EMV v4.3 Book 2- 6.5 Dynamic Data Authentication (DDA) - p66
 * EMV v4.3 Book 2- 6.5.2 Dynamic Signature Verification - p68
 * EMV v2.2 Book C-3 - A.2 Data Elements by Name - p97
 * EMV v2.2 Book C-3 - 2.4.7 Online Processing (EMV Mode) - p14
 * EMV v2.2 Book C-3 - 2.4.8 Completion (EMV Mode) - p15
 * EMV v2.2 Book B - 3.3 Combination Selection - p24
 *
 * Additional References and coding comments
 * EMV Contactless Book A - 5.8.2 Application Selection and Kernel Activation
 * EMV Contactless C-3 - 2.4 New Transaction Processing Sequence p12
 * EMV Contactless C-3 - 5.2 Initiate Application Processing p41
 * EMV Contactless C-3 - C.1 Dynamic Signature Verification p132
 * EMV Contactless C-3 - Figure C-1: Fast DDA (fDDA) EMV Mode Example p134
 * EMV v4.2 Book 2 - 10.1 Initiate Application Processing p92
 * EMV v4.2 Book 2 - 10.2 Read Application Data p95
 * EMV v4.2 Book 3 - 10.3 Offline Data Authentication (AFL & AIP operation)
 * Describes the AFL & AIP data is to be used in the validation of Signed Data.
 * EMV Contactless Book C-3 - 2.4 New Transaction Processing Sequence p12
 * EMV v4.2 Book 3 - 10.3 Offline Data Authentication (AFL & AIP operation)
 * The above 2 references indicate that the AFL & AIP data is to be used in the
 * validation of the Hash of the signed data, HOWEVER Visa fDDA only works if
 * the AFL and AIP are left empty.
 * @return Transaction Outcome Code
 */
@Override
protected Outcome EMVTransaction()
{
    Log.Write("UML 8.0 PopulatePDOL()", Log.PROTOCOL);
    // UML 8.0 PopulatePDOL()
    // The Processing Options Data Objects List (PDOL) is the list of data elements that
```

```
    // the card requires to approve the transaction.  The terminal populates the PDOL
    // with data, the data elements would usually include transaction amount, currency,
    // unpredictable number, transaction date.
    // EMV v2.2 Book B - 3.6 Data Element Processing - p34
    // EMV v2.2 Book C-3 - 2.4.1 Initiate Application Processing - p12
    // EMV v2.2 Book C-3 - 5.1 Initiate Application Processing - p40
    // EMV v4.3 Book 3 - 5.4 Rules for Using a Data Object List (DOL) – p38
    // EMV v4.3 Book 4 - Annex A - Coding of Terminal Data Elements - p115
    // EMV v4.3 Book 3 - Annex C3 Cardholder Verification Rule Format – p163
    // EMV v2.2 Book C-3 - A.2 Data Elements by Name (9F66) - p110
    // EMVv2.2  Book C-3 - Annex C Fast Dynamic Data Authentication - p127
    byte[] dol = this.PopulatePDOL();
    Log.Write("UML 9.0 GetProcessingOptions(PDOL) ", Log.PROTOCOL);
    // UML 9.0 GetProcessingOptions(PDOL data)
    // In the Visa fDDA transaction Get Processing Options (GPO) is used to request completion
    // of the transaction.  The PDOL data must contain all of the data elements requested by
    // the card otherwise the transaction will be rejected
    // EMV v2.2 Book C-3 - 2.4.1 Initiate Application Processing - p12
    // EMV v2.2 Book C-3 - 5.2 Initiate Application Processing - p40
    // EMV v2.2 Book C-3 - 5.2.1 Get Processing Options (GPO) Command - p40
    // EMV v2.2 Book C-3 - 5.2.2 Initiate Application Processing - p40 to p46
    // EMV v4.3 Book 3 - 6.5.8.4 Data Field Returned in the Response - p60
    ResponseAPDU response = this.Reader.GetProcessingOptions(dol);
    if (response.getSW() == Const.SW_SUCCESS)
    {
       // Split the HEX string response from the card into individual fields with a TAG and Value
       // EMV v4.3 Book 3 - Annex B Rules for BER-TLV Data Objects p155
       // Or if it isn't TLV decode it as a Format 1 object –
       // EMV v4.3 Book 3 - 6.5.8.4 Data Field Returned in the Response Message
       if(!this.CardData.DecodeResponse(response))
          this.CardData.FormatGPOResponse(response.getData());
       byte [] iad = this.CardData.FindData(Const.TAG_IAD);
       Log.Write("UML 9.7 Transaction Approved", Log.PROTOCOL);
       // UML 9.7 Transaction Approved
       // Application Cryptogram (AC), AFL
       // If the card approves the completion of the transaction in offline mode, it will
       // return Transaction Cryptogram  (TC) in the Cryptogram Information Data (CID).
       // The card also returns all of the data elements required by the terminal to
       //  complete the transaction:
       // Signed Dynamic Application Data (SDAD) used by the terminal to verify that the
       // card has approved the same transaction that the terminal sent.
       // Application Cryptogram (AC) used in the completion of the transaction with the
       // Bank to validate that a valid card completed the transaction.
       // Application File Locator (AFL) contains the location in the card's file
       // structure where the terminal can read the data elements required to complete
       // the transaction.
       // EMV v2.2 Book C-3 - 5.4.3 Determine the Card Disposition - p50
       // EMV v2.2 Book C-3 - 5.2.2.2 GPO Response SW1 SW2 - p43
       // EMV v2.2 Book C-3 - 5.2.2.3 Contactless Path Determination - p46
       // EMV v2.2 Book C-3 - A.2 Data Elements by Name - p97
       // EMVv2.2  Book C-3 - Annex C Fast Dynamic Data Authentication - p127
       // EMV v2.2 Book C-3 - C.1 Dynamic Signature Verification - p128
       // EMV v4.3 Book 3 - 6.5.8 Get Processing Options APDUs - p59
       // EMV v4.3 Book 3 - 6.5.8.4 Data Field Returned in the Response - p60
       // TC Returned by Get Processing Options (9F10)
       if (Util.BitCompare(iad[4], Const.IAD_VISA_STATUS_MASK, Const.IAD_VISA_TC))
       {
          Log.Write("--------- TC Returned by Get Processing Options (9F10) -------", Log.PROTOCOL);
          Log.Write("UML 10.0 ReadAFLRecord(SFI, Record)", Log.PROTOCOL);
          // UML 10.0 ReadAFLRecord(SFI, Record)
          // The Application File Locator (AFL) returned in the GPO response contains
          // the location of the data records which contain the data elements that
          // the terminal will require to complete the transaction.  The AFL contains
          // a range of SFI / Records.  The data the AFL points to will include the
          // public keys required to decrypt the SDAD returned in the GPO response.
          // EMV v2.2 Book C-3 - 5.3 Read Application Data - p47
          // EMV v4.3 Book 3 - 10.2 Read Application Data - p93
          // EMV v4.3 Book 3 - 6.5.11 Read Record Command-Response APDUs - p65
          // UML 10.1 AFL record
          // The card will return the requested SFI / Record
          // EMV v4.3 Book 3 - 10.2 Read Application Data - p93
          // EMV Contactless Book C-3 - 2.4.2 Read Application Data p13
          byte[] afl_data = this.GetAFLData(); // Fields already added to CardData
          byte[] sda_data = this.GetSDAData();
```

```
            Log.Write("UML 11.0 ValidateTransaction()", Log.PROTOCOL);
            // UML 11.0 ValidateTransaction()
            // If the card has approved the transaction Offline Data Authentication
            // is performed by the terminal to verify the dynamic signature and
            // authenticate the data returned by the card in response to the GPO
            // command.  The terminal verifies the DDA dynamic signature contained in
            // the SDAD using the CA public key, issuer public key and the card's
            // public key.  The resulting data packet is a SHA-1 hash of the terminal
            // unpredictable number, the card unpredictable number, the transaction
            // amount, the currency and the date.  This process ensures that the card
            // has verified the transaction without altering any of the values in the
            // transaction.  The unpredictable numbers are used to ensure that
            // transactions cannot be recorded and replayed.  If the signed data in
            // the SDAD passes the validation  the transaction is deemed valid and
            // the terminal will approve the transaction.
            // EMV v2.2 Book C-3 - 5.6 Offline Data Authentication - p58
            // EMV v4.3 Book 2 - 6 Offline Dynamic Data Authentication - p51
            // EMV v4.3 Book 2- 6.2 Retrieval Certification Authority Public Key - p60
            // EMV v4.3 Book 2- 6.3 Retrieval of Issuer Public Key - p60
            // EMV v4.3 Book 2- 6.3 Retrieval of ICC Public Key - p63
            // EMV v4.3 Book 2- 6.5 Dynamic Data Authentication (DDA) - p66
            // EMV v4.3 Book 2- 6.5.2 Dynamic Signature Verification - p68
            byte[] sdad = this.CardData.FindData(Const.TAG_SDAD); // TAG_SDAD 0x9F4B
            if (sdad.length > 0)
            {
                // EMV Contactless C3 – C.1 Dynamic Signature Verification - p132
                // EMV Contactless C3 – Table C-1: Terminal Dynamic Data for Input to DDA Hash
Algorithm - p132
                byte[] ddol = this.TerminalTags.FindData(Const.TAG_UNPREDICTABLE_NUMBER);
                ddol = Util.ByteInsert(ddol, this.TerminalTags.FindData(Const.TAG_TRANSACTION_AMOUNT),
-1);
                ddol = Util.ByteInsert(ddol, this.TerminalTags.FindData(Const.TAG_CURRENCY_CODE), -1);
                ddol = Util.ByteInsert(ddol, this.CardData.FindData(Const.TAG_CARD_AUTH_DATA), -1);
                Log.Write("DDOL => " + Util.HexString(ddol), Log.PROTOCOL);
                Log.Write("---------------------- Validating SDAD -----------------------",
Log.PROTOCOL);
                Log.Write("9F4B => " + Util.HexString(sdad), Log.PROTOCOL);
                // Generate the Issuer Public Key from the CA Public Key
                // EMV v4.3 Book 2- 6.3 Retrieval of Issuer Public Key - p60
                if (GenerateIssuerMod())
                {
                    // Generate the ICC Public Key from the Issuer Public Key
                    // EMV v4.3 Book 2- 6.3 Retrieval of ICC Public Key - p63
                    if (GenerateICCMod(afl_data, sda_data))
                    {
                        // UML 11.0 ValidateTransaction()
                        // Decrypt and validate the SDAD using the ICC Public key
                        // EMV v4.3 Book 2- 6.5 Dynamic Data Authentication (DDA) - p66
                        // EMV v4.3 Book 2- 6.5.2 Dynamic Signature Verification - p68
                        byte[] icc_dynamic_number = this.Crypto.DynamicSignatureVerification(ICCMod,
ICCExp, sdad, ddol);
                        if (icc_dynamic_number.length > 0)
                        {
                            // Insert the result into Terminal TAG_ICC_DYNAMIC_NUM 9F4C
                            this.TerminalTags.Update(Const.TAG_ICC_DYNAMIC_NUM, icc_dynamic_number);
                            // EMV Contactless C-3 - 2.4.8 Completion (EMV Mode)
                            outcome = Outcome.APPROVED;
                        }
                    }
                }
            }
            if (outcome == Outcome.APPROVED)
            {
                Log.Write("Application Cryptogram " +
Util.HexString(this.CardData.FindData(Const.TAG_APPLICATION_CRYPTOGRAM)), Log.PROTOCOL);
                Log.Write("------------------ TRANSACTION SUCCESSFUL --------------------",
Log.PROTOCOL);
            }
            else
            {
                Log.Write("------------------- SDAD CRYPTOGRAM ERROR --------------------",
Log.PROTOCOL);
            }
        }
```

```
        else Log.Write("ERROR - Kernel3.DoTransaction() - Signed Dynamic Data Missing", Log.ERROR);
    }
    // Check if PDOL returned AAC (Application Authentication Cryptogram)
    else if (Util.BitCompare(iad[4], Const.IAD_VISA_STATUS_MASK, Const.IAD_VISA_AAC))
    {
        Log.Write("-------- ACC Returned by Get Processing Options (9F10) -------", Log.PROTOCOL);
        Log.Write("UML 9.9 Transaction Declined - AAC ", Log.PROTOCOL);
        // UML 9.9 Transaction Declined - AAC
        // If the card declines the transaction it returns AAC in the CID.  The card declines the
        // transaction when it cannot be completed as requested by the terminal.
        // EMV v2.2 Book C-3 - 5.4.3 Determine the Card Disposition - p50
        // EMV v2.2 Book C-3 - A.2 Data Elements by Name -= p97
        // EMV v2.2 Book C-3 - 2.4.7 Online Processing (EMV Mode) - p14
        // EMV v2.2 Book C-3 - 2.4.8 Completion (EMV Mode) - p15
        outcome = Outcome.DECLINED;
        Log.Write("------------------- TRANSACTION DECLINED ---------------------", Log.PROTOCOL);
    }
    // Check if PDOL returned ARQC (Authorisation Request Cryptogram)
    else if (Util.BitCompare(iad[4], Const.IAD_VISA_STATUS_MASK,  Const.IAD_VISA_ARQC))
    {
        Log.Write("------- ARQC Returned by Get Processing Options (9F10) -------", Log.PROTOCOL);
        Log.Write("UML 9.8 Transaction Must Go Online –ARQC", Log.PROTOCOL);
        // UML 9.8 Transaction Must Go Online –ARQC
        // If the card requires online completion of the transaction it will return ARQC in the
        // Cryptogram Information Data (CID).  Online completion is required when the amount of the
        // transaction exceeds the cards offline transaction limit or offline cumulative limit or
        // when the number of offline transactions exceeds the number of consecutive offline
        // transactions.
        // EMV v2.2 Book C-3 - 5.4.3 Determine the Card Disposition - p50
        // EMV v2.2 Book C-3 - A.2 Data Elements by Name - p97
        // EMV v2.2 Book C-3 - 2.4.7 Online Processing (EMV Mode) - p14
        // EMV v2.2 Book C-3 - 2.4.8 Completion (EMV Mode) - p15
        outcome = Outcome.REQUEST_ONLINE;
        Log.Write("---------------- TRANSACTION MUST GO ONLINE ------------------", Log.PROTOCOL);
    }
    else // PDOL returned invalid IAD status code
    {
        Log.Write("--- Invalid IAD returned by Get Processing Options (9F10) ----", Log.PROTOCOL);
        Log.Write("UML 9.10 Command Error ", Log.PROTOCOL);
        // UML 9.10 Command Error
        // The possible error codes returned by Get Processing Options are:
        // 6984- Try Another Interface.  The transaction should be reattempted using either the
        // contact interface or the magnetic stripe interface.
        // 6985 - Select Next.  The transaction should be reattempted using the next combination
        // of Kernel / AID (if any).
        // 6986 - Try Again, reattempt the transaction with the same parameters.  The card must
        // be represented, this may occur if the card is removed too early.
        // EMV v2.2 Book C-3 - 5.2.2.2 GPO Response SW1 SW2 - p43
        // EMV v2.2 Book C-3 - 2.4.8 Completion (EMV Mode) - p15
        // EMV v2.2 Book B - 3.3 Combination Selection - p24
        Log.Write("IAD status code (9F10) : " + Util.HexString(iad), Log.ERROR);
        Log.Write("------------------- TRANSACTION DECLINED ---------------------", Log.PROTOCOL);
    }
    return outcome;
}

};
```

A.4 Protocol Emulator Trace Log – Kernel 3 fDDA Contactless Transaction

```
[24-05-2015 18:58:14.164] Card Reader Type Selected: PC/SC terminal ACS ACR1222 3S PICC Reader PICC
0
[24-05-2015 18:58:20.388] EntryPoint: Set Interface to NFC
[24-05-2015 18:58:20.419] ---------------------- Wait for Card -----------------------
[24-05-2015 18:58:27.65] --------------------- Card Detected ----------------------
[24-05-2015 18:58:27.252] ATR: 19 bytes => 3B8E800180318066B08412016E018300900003
[24-05-2015 18:58:27.252] Historical Bytes => 80318066B08412016E0183009000
[24-05-2015 18:58:27.268] Hash Code => 1DFB2774
[24-05-2015 18:58:27.268] Card Name => Contactless Barclaycard Visa
[24-05-2015 18:58:27.268] sendAPDU => 80CA9F7F00
[24-05-2015 18:58:27.299] respAPDU <=
9F7F2A4090636112918101010002006BEFAC5ECDE4F10022006419320064194200600OC0059424531570000FFFF {9000}
[24-05-2015 18:58:27.299] ICC Maker / Type => 0x4090 / 0x6361

[24-05-2015 18:58:27.299] ------------------- Start Transaction ---------------------
[24-05-2015 18:58:27.299] UML (kernel) 4.0 ListAvailableApplications()
[24-05-2015 18:58:27.315] CardReader: PPSE => 2PAY.SYS.DDF01
[24-05-2015 18:58:27.315] sendAPDU => 00A404000E325041592E5359532E4444463031
[24-05-2015 18:58:27.361] respAPDU <=
6F20840E325041592E5359532E4444463031A50EBF0C0B61094F07A0000000031010 {9000}
[24-05-2015 18:58:27.361] sendAPDU => 00B2010C00
[24-05-2015 18:58:27.377] respAPDU <=  {6A82}
[24-05-2015 18:58:27.377] UML (kernel) 4.1 List of Available Applications
[24-05-2015 18:58:27.377] EntryPoint: Kernel ID => 3
[24-05-2015 18:58:27.377] ---------------------- Kernel3 Selected ----------------------
[24-05-2015 18:58:27.377] UML (kernel) 7.0 SelectApplication(A0000000031010)
[24-05-2015 18:58:27.393] sendAPDU => 00A4040007A0000000031010
[24-05-2015 18:58:27.439] respAPDU <=
6F3D8407A0000000031010A532501042415243 4C4159434152442056495341 9F38189F66049F02069F03069F1A0295055F2A
029A039C019F37045F2D02656E {9000}
[24-05-2015 18:58:27.439] UML (kernel) 7.1 Trasnaction Setup data inc PDOL
[24-05-2015 18:58:27.439] UML (kernel3) 8.0 PopulatePDOL()
[24-05-2015 18:58:27.455] UML (kernel3) 9.0 GetProcessingOptions(PDOL)
[24-05-2015 18:58:27.455] sendAPDU =>
80A800002383213A208000000000000010000000000000008260000000000082615052400179C6A9E
[24-05-2015 18:58:27.767] respAPDU <=
7781C29F4B8180184F20039A940B5D7D887EBC88ADE42A95A6042429FF0988EF25AC0FD98BF571F98B51E8039C4100AD0251
9348A5A7617D412148F3A95E0BFD82CF267978ACA9398263DA86E6FF9327325C7253510F4E28918126DB3719FADC23A66E3D
589FB6D4B9DF0718B0B7B1859F12095BDB65FF0FC81C59315003C50C6359356660E9519404100203009F1007060B0A039000
0057134929xxxxxxxx4037D161120103635000000001F820220009F360201C39F2608FCF9E2AC71DE56219F6C021000
{9000}
[24-05-2015 18:58:27.767] ---------------------- EMV Mode Processing --------------------
[24-05-2015 18:58:27.767] UML (kernel3) 9.7 Transaction Approved -
TC+SDAD+ApplicationCryptogram(AC)+AFL
[24-05-2015 18:58:27.767] --------- TC Returned by Get Processing Options (9F10) -------
[24-05-2015 18:58:27.767] UML (kernel3) 10.0 ReadAFLRecord(SFI, Record)
[24-05-2015 18:58:27.767] sendAPDU => 00B2021400
[24-05-2015 18:58:27.876] respAPDU <=
7081C08F010790819066658CBCE38531130ADC606433713E2FCBCA77F8C07E222D08B61B72C49168B8A64A0BF77D72E896F6
BCD1D5386605C0798C55EEB14481F5ED3103296DD90D1F06D67EFB7DAA26D18960FD74B13862A9B26F9EAF61A1BB55000440
128547B9C05D262DDBD64A3FB53CFA54DE8FF17AFCBEA65C5BF8ED1BF610F05E8D90AB9CEFBB7670A9F30A010E115B51F9F5
E85B599224313E45081DF97177B20BBB1C8505393FEB07C81E3A6AD4E9F6FDDFE23E6D323763D560E99F320103 {9000}
[24-05-2015 18:58:27.876] sendAPDU => 00B2031400
[24-05-2015 18:58:28.1] respAPDU <=
7081D19F46819027F4178D281BFC2A3CA7E0EFAE6FA4515848FA27940DAF58B61D66D1487901806B286944C3489C26506772
D9339EC6C2A8797BA6E35D71D30BDF5B8F3C35F2B881263B0C0A1E4A755F75862AB46C62BE5E80537774C6F8FF592B066FFF
FAA3900525DA711DFA0A7537EE686AB3AF77A0971D39D530C1C36CA5468EF32DEADA510F1621E2F723473804A9F12CD7D9EC
C49F4701039F481A3B332E4D800610E88F140F0E0F5B073D17F92073BA3360E88EC55A084929xxxxxxxx40375F3401305F24
031611309F69050151738D3A {9000}
[24-05-2015 18:58:28.17] EMVTerminal.GetSDAData()
[24-05-2015 18:58:28.17] UML (kernel3) 11.0 ValidateTransaction()
[24-05-2015 18:58:28.17] DDOL => 179C6A9E00000000010008260151738D3A
[24-05-2015 18:58:28.17] ---------------------- Validating SDAD ----------------------
[24-05-2015 18:58:28.17] 9F4B =>
184F20039A940B5D7D887EBC88ADE42A95A6042429FF0988EF25AC0FD98BF571F98B51E8039C4100AD02519348A5A7617D41
2148F3A95E0BFD82CF267978ACA9398263DA86E6FF9327325C7253510F4E28918126DB3719FADC23A66E3D589FB6D4B9DF07
18B0B7B1859F12095BDB65FF0FC81C59315003C50C6359356660E951
[24-05-2015 18:58:28.32] ---------------------- Recovered Issuer Key ----------------------
[24-05-2015 18:58:28.32] Header      6A
[24-05-2015 18:58:28.32] Cert Format 02
```

```
[24-05-2015 18:58:28.32] Issuer ID   492949FF
[24-05-2015 18:58:28.32] Cert Expiry 1217
[24-05-2015 18:58:28.32] Cert Serial 014DC2
[24-05-2015 18:58:28.32] Hash Algo   01
[24-05-2015 18:58:28.32] Key Algo    01
[24-05-2015 18:58:28.32] Key Length  90
[24-05-2015 18:58:28.32] Exp Length  01
[24-05-2015 18:58:28.32] ICC Key
C639B9467B9E56D9C6395E6B934CD2ACFBEC953C6CE8D31923130F2046384EFD5BDE921C3ED12505B1BFE7DDF488E62CA475
AAFC1A6919C944F658BA0230C4894CA5E5E2AA0262B09A5B3CB7ECB6CD93AF0D717C40642E5B14B070B0BE7181CECDB79396
C89C36414C8EF21D
[24-05-2015 18:58:28.32] Hash        8652C1B69B5255D28CB7C2F741873D77FC8E52B3
[24-05-2015 18:58:28.32] Trailer     BC
[24-05-2015 18:58:28.32] ----------------------- Recovered Issuer Key -----------------------
[24-05-2015 18:58:28.32] CryptographyRSA.RetrieveIssuerPublicKey() - SUCCESSFUL
[24-05-2015 18:58:28.48] ------------------------ Recovered ICC Key ------------------------
[24-05-2015 18:58:28.48] Header      6A
[24-05-2015 18:58:28.48] Cert Format 04
[24-05-2015 18:58:28.48] PAN         4929xxxxxxxx4037FFFF
[24-05-2015 18:58:28.48] Cert Expiry 1116
[24-05-2015 18:58:28.48] Cert Serial 3A6B02
[24-05-2015 18:58:28.48] Hash Algo   01
[24-05-2015 18:58:28.48] Key Algo    01
[24-05-2015 18:58:28.48] Key Length  80
[24-05-2015 18:58:28.48] Exp Length  01
[24-05-2015 18:58:28.48] ICC Key
D95D3658164C174E64056D72B2B34147BA26D7AADAC5731B01F52E7799FA099278748CB99424D906DA2F9E59D86BC8BD72C3
2E5DF772A4DC9F0149C2DB82B41374A7EC4BD1560CFA9B42646AC8394ECF85B573C7AC399D22CCA4429D9442CA02EF83122C
5D91
[24-05-2015 18:58:28.48] Hash
[24-05-2015 18:58:28.48] Trailer     BC
[24-05-2015 18:58:28.48] ------------------------ Recovered ICC Key ------------------------
[24-05-2015 18:58:28.48] CryptographyRSA.RetrieveICCPublicKey() - SUCCESSFUL
[24-05-2015 18:58:28.48] CryptographyRSA.DynamicSignatureVerification()
[24-05-2015 18:58:28.48] CryptographyRSA.DynamicSignatureVerification() - SUCCESSFUL
[24-05-2015 18:58:28.48] -------------------------- Recovered SDAD -------------------------
[24-05-2015 18:58:28.48] Header    6A
[24-05-2015 18:58:28.63] Format    05
[24-05-2015 18:58:28.63] Algorithm 01
[24-05-2015 18:58:28.63] Length    03
[24-05-2015 18:58:28.63] Data      0201C3
[24-05-2015 18:58:28.63] Padding
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
[24-05-2015 18:58:28.63] Hash      EB8880140A352EAD82370FA47F6F4F52623068F9
[24-05-2015 18:58:28.63] Trailer   BC
[24-05-2015 18:58:28.63] -------------------------- Dynamic Data --------------------------
[24-05-2015 18:58:28.63] Length         02
[24-05-2015 18:58:28.63] Dynamic Number 01C3
[24-05-2015 18:58:28.63] -------------------------- Dynamic Data --------------------------
[24-05-2015 18:58:28.63] Application Cryptogram FCF9E2AC71DE5621
[24-05-2015 18:58:28.63] ------------------ TRANSACTION SUCCESSFUL --------------------
[24-05-2015 18:58:28.63]
-------------------- TRANSACTION Approved ---------------------
Application  A0000000031010 - BARCLAYCARD VISA
Amount       000000000100
Currency     0826
Date         150524
-------------------- TRANSACTION Approved ---------------------
```

A.5 Abstract Model – Kernel 3 fDDA Contactless Transaction

FS.SDAD.Algorithms

cardAuthRelData [9, p.90] *encrypt [9, p.134]*
hashCode [9, p.132], [2, p.68]

To create an *SDAD*, the card needs a few algorithms to be available. They are abstracted away here as functions from the needed input, coming from a *PDOL* or the card state, to the actual values needed for the terminal. Function *cardAuthRelData* generates a unpredictable number given a card id *CID*. Similarly, the *hashcode* function produces a hash value for the inputs: card unpredictable number, the *PDOL*'s unpredictable number, the *PDOL* amount, and the *PDOL* currency. Finally, all that gets encrypted using the card's transaction counter paired with this hashcode and the card's private key.

$$
\begin{array}{l}
cardAuthRelData : CID \to \mathbb{N} \\
hashCode : \mathbb{N} \times (\mathbb{N} \times (\mathbb{N} \times CURRENCY)) \to \mathbb{N} \\
encrypt : \mathbb{N} \times (\mathbb{N} \times SDAD) \to \mathbb{N} \\
decrypt : \mathbb{N} \times (\mathbb{N} \times \mathbb{N}) \to SDAD \\
\hline
\forall\, trCnt, puKey, prKey, issuerKey, cypher : \mathbb{N} \bullet \\
cypher = encrypt\,(trCnt, (prKey, decrypt\,(issuerKey, (puKey, cypher)))) \\
\forall\, trCnt, puKey, prKey, issuerKey : \mathbb{N};\ msg : SDAD \bullet \\
msg = decrypt(issuerKey, (puKey, encrypt\,(trCnt, (prKey, msg))))
\end{array}
$$

theorem rule lEncryptResType

$$\forall\, trCnt, prKey : \mathbb{N};\ msg : SDAD \bullet encrypt\,(trCnt, (prKey, msg)) \geq 0$$

theorem rule lCardAuthDataResType

$$\forall\, cid : CID \bullet cardAuthRelData\,(cid) \geq 0$$

theorem rule lHashCodeResType

$$
\begin{array}{l}
\forall\, cUpno, pdolUpno, pdolAmnt : \mathbb{N};\ pdolCurrency : CURRENCY \bullet \\
\quad hashCode\,(cUpno, (pdolUpno, (pdolAmnt, pdolCurrency))) \geq 0
\end{array}
$$

We create an *FDDA NFC SDAD* output, providing apprpropriate *PDOL* with corresponding message payload. We also need to check the card information is correct with respect to the card's requested *PDOL* data responded by the terminal.

$$
\begin{array}{l}
FDDANFCVisaSDAD \\
\hline
NFCVisaPDOL?;\ Card?;\ NFCVisaSDAD! \\
\hline
cstart? \leq_D pdolDate? \leq_D cend? \\
pdolAmount? \leq climit? \\
pdolAmount? \leq nfcOfflineCounter? \\
0 < contactCounter? \\
\\
sdadCardTrCounter! = transacCounter? \\
sdadHashCode! = hashCode\,(cardAuthRelData\,(cid?), \\
\qquad (pdolUpno?, (pdolAmount?, pdolCurrency?))) \\
\hline
\end{array}
$$

These include the card start/end date with respect to the transaction date, the card limit and the transaction amount, the card conctactless (*NFC*) counter threshold (*i.e.* when it reaches zero, we need to go online). For the *SDAD*, we need to encode the current transaction counter from the card, as well as a hashcode with information relating the card *CID*-based unpredictable number, and the corresponding unpredictable number, amount and currency from the terminal's *PDOL* response.

Figure 43 - Example Section of Z Abstract Model

The Z abstract model is 90 pages and therefore too large to be added as an appendix to this thesis. The Z abstract model can be read in the technical report Freitas and Emms 2014 [61] which contains the Z specification and the proofs.