

e-EMV: Emulating EMV for Internet payments using Trusted Computing technology

Shane Balfe and Kenneth G. Paterson

Technical Report
RHUL-MA-2006-10
17 November 2006



Department of Mathematics
Royal Holloway, University of London
Egham, Surrey TW20 0EX, England
<http://www.rhul.ac.uk/mathematics/techreports>

Contents

1	Introduction	3
2	System model	7
2.1	Threats	10
3	Card payments and the Internet	11
3.1	SSL	12
3.2	SET	12
3.3	3-D Secure, Verified by Visa and SecureCode	13
3.4	Summary	15
4	EMV	15
4.1	Transaction processing	16
4.2	EMV Key Usage	18
4.2.1	SDA and DDA	18
4.2.2	Application Cryptograms	19
4.2.3	Cardholder Verification	19
5	Trusted Computing	20
5.1	TPM Specification	20
5.1.1	Protected capabilities and shielded locations	20
5.1.2	Integrity Measurement and Storage	21
5.1.3	TPM Keys and Key Usage	22
5.1.4	Reporting and Attestation	23
5.1.5	Delegation and Migration	25
5.1.6	TCG Credentials	26
5.2	Trusted PC and Trusted Server Specifications	27
5.3	Trusted Server Specification	27
5.4	TNC Specification	27
5.5	OS and Processor Support	28
6	An overview of e-EMV	29
6.1	Enrollment	30
6.2	Transaction Architecture	32
7	Installing and Instantiating an e-EMV card	33
7.1	Account Activation	34
7.1.1	Privacy CA Approach	34
7.1.2	DAA Approach	35
7.2	Secure Application Delivery	36

8	e-EMV in Operation	39
8.1	Customer - Merchant Interaction	39
8.2	Application Cryptograms	43
9	An Application of Trusted Computing to CNP transactions	44
9.1	Pseudonymous Credit Cards	44
10	Conclusions	45
11	Appendices	46
A	CMK Generation and e-EMV Migration	46

Abstract

The introduction of Static Data Authentication (SDA) compliant EMV cards with their improved cardholder verification and card authentication capabilities has resulted in a dramatic reduction in the levels of fraud seen at Point of Sale (POS) terminals. However, with this POS-based reduction has come a corresponding increase in the level of fraud associated with Internet-based Card Not Present (CNP) transactions. This increase is largely attributable to the fact that Internet-based CNP processing has no easy way of integrating EMV into its transaction architecture. In this regard, payment is reliant on Mail Order Telephone Order (MOTO) based processing where knowledge of card account details is deemed a sufficient form of transaction authorisation.

This report aims to demonstrate how Trusted Computing technology can be used to emulate EMV for use in Internet-based CNP transactions. Through a combination of a Trusted Platform Module, processor (with chipset extensions) and OS support we show how we can replicate the functionality of standard EMV-compliant cards. The usage of Trusted Computing in this setting allows a direct migration to more powerful Combined DDA and application cryptogram generation (CDA) cards as well as offering increased security benefits over those seen in EMV's deployment for POS transactions. Customer to Merchant interaction in our setting mirrors transaction processing at traditional POS terminals. We build upon the services offered by Trusted Computing in order to provide a secure and extensible architecture for Internet-based CNP transactions.

1 Introduction

Over the past ten to fifteen years the Internet has been transformed from a research-oriented tool into a global platform for electronic commerce. With this transformation, the use of one particular method of Internet-based payment has emerged as the predominant means through which on-line goods are purchased. This particular form of payment, typically referred to as Card Not Present¹ (CNP) transactions, uses the account information embossed on physical (debit/credit) payment cards to purchase items on-line. Unfortunately, CNP transactions whilst ubiquitous, are currently far from secure.

A recent report by the Association for Payment Clearing Services (APACS) on card fraud [4] showed that Internet-based CNP transactions and their as-

¹For the remainder of this paper all references to CNP transactions refer to Internet-based CNP transactions.

sociated chargebacks² accounted for nearly 27% of all card fraud perpetrated in 2005 in the UK. This translated into £117 million in losses for card issuers and merchants.

This proliferation of Internet-based commerce (and the increasing level of fraud associated with it) has resulted in a great deal of effort in developing protocols for securing CNP transactions. However, very few of these protocols have enjoyed any quantifiable success. Although some innovative schemes such as Paypal have enjoyed some success (largely due to a tie in with eBay), the vast majority of Internet-based CNP payments use a single protocol suite, namely SSL, to secure card account information in transit.

Unfortunately, this usage of SSL is not a panacea for enabling secure CNP transactions. SSL was not designed as a payment protocol but instead adopted as a de facto standard for securing CNP transactions. Indeed, the use of SSL in CNP transactions has a number of short-comings. These ‘flaws’ in SSL can largely be attributed to the marriage of convenience that exists with current CNP-based card processing and are not necessarily intrinsic to the protocol itself. In securing CNP transactions, SSL is used only in relation to securing the payment channel over which customer card account details are sent, everything else remains outside of SSL’s protection remit. This poses two fundamental problems for a CNP payment architecture. Firstly, from the merchant’s perspective: there is no guarantee that the customer owns the account number being proffered in a particular payment transaction. Secondly, from the customer’s perspective: there is little, if any, assurance that the merchant will endeavour to protect sensitive cardholder data stored on their servers. In this regard, transaction processing is reliant on a Mail Order Telephone Order (MOTO) based system whereby demonstrating knowledge of a card’s Personal Account Number (PAN) and corresponding Card Security Code (CSC) are deemed a sufficient form of transaction authorisation.

The real world roll-out of EMV and the accompanying desire to replicate the fraud reduction seen in Card Present (CP) transactions has resulted in a number of proposals to utilise EMV’s functionality for Internet-based payments [29, 32, 10, 1, 38, 35]. There are two main advantages to these proposals. Firstly, by integrating EMV into a CNP transaction, the cardholder can prove that the card is in their physical possession. Secondly, only the valid owner of a card can authorise a transaction, as only the card owner should be capable of demonstrating that they know the EMV card’s secret PIN. However, these approaches haven’t seen any real traction in the market place. A possible explanation for this is the underlying assumption that

²A chargeback is a term used to refer to the situation in which a genuine cardholder reports an unknown and possibly fraudulent transaction to their card issuer.

users would make use of card readers connected to their PC platforms. This assumption engenders an additional cost in the form of distributing card readers to end-users. Unfortunately, even if the cost issue could be surmounted, these approaches alone offer only limited security gains. This is due to the lack of a trusted path between the card-reader and host, as well as a lack of Operating System support for application isolation. Without these, a piece of malicious software could passively observe PIN entries, actively modify transaction data and possibly generate new transactions, enabling criminals to remotely take control of users' credit/debit cards. As highlighted in a recent IBM report, the threat from malware is increasing at an unprecedented rate [39]. Customised malware attacks and 'spear phishing' (directed phishing attacks) are starting to target specific industries and organisations and as such, present a particular challenge to the future of Internet-based CNP transactions. There has, however, been a recent development with respect to integrating EMV into CNP transactions that aims to indirectly address the malware issue with the proposed use of "unconnected" card readers [35]. Unfortunately, this approach also suffers with respect to the additional costs associated with distributing card readers to end-users and once deployed cannot be updated to address new threats as they emerge. Furthermore, the exact details of how these readers would operate remains unclear as there are currently no publicly available specifications detailing their precise operation.

This report forms an extension to earlier work [8] in proposing a new security architecture for securing CNP transactions through the software emulation of EMV (herein referred to as e-EMV) running on Trusted Platforms. Here emulation of EMV-compliant cards confers "tamper resistant" properties normally associated with physical EMV-compliant cards. This makes it possible to demonstrate card ownership and authentication as per EMV's SDA/DDA/CDA approaches in a virtualised environment. Additionally, by creating an environment where EMV transaction flows can be mapped directly to e-EMV transactions we can avoid any expensive re-engineering of the back-end financial network. Throughout the rest of this report we demonstrate how our approach can achieve the following:

1. A merchant can ensure that a customer claiming to present a particular e-EMV card is the legitimate owner of that card.
2. Issuing and acquiring banks can authenticate customers' transaction data to prevent authorisation of illegitimate payment requests.
3. A merchant can obtain a payment guarantee by being able to demonstrably show authorisation of a transaction by a customer.

4. A customer can be assured that the merchant with whom they are engaging in a transaction will endeavor to protect sensitive cardholder information prior to transaction initiation. Additionally, a customer may be assured that parties privy to transaction-related data such as the issuer, the acquirer or particularly third party processing facilities satisfy pre-described security and privacy requirements prior to obtaining cardholder data.
5. A customer can securely port their e-EMV card from one Trusted Platform to another.
6. An executing e-EMV application can avoid the threat posed by malware by hosting e-EMV cards in their own isolated memory partition, free from observation and interference.

Additionally, whilst Static Data Authentication (SDA) remains a popular choice for POS EMV cards, e-EMV cards do not have the same cost restrictions associated with their deployment. The popularity of SDA for POS transactions is largely due to its utilisation of cheaper chip technology compared to that of Dynamic Data Authentication (DDA) and Combined DDA and application cryptogram generation (CDA), which have additional costs associated with the increased chip complexity. In certain deployments it is more economical to accept a higher level of risk than to migrate directly to a more expensive card. This is not the case with e-EMV.

The use of Trusted Computing in our setting allows for a direct migration to DDA/CDA-compliant cards. This is because the price of the physical medium, at least from the card issuer's perspective, would essentially be zero. Additionally, Trusted Computing platforms will not have the same processing constraints in place as integrated chip cards, and can thus natively support DDA/CDA capabilities. In this respect, the use of DDA/CDA allows the merchant to determine whether or not the emulated card is genuine, as well as being able to determine if the data personalised to the card has altered since personalisation. An additional feature of our approach is a degree of future-proofing inherent to our architecture. Updates to the EMV specifications can be remodeled as software processes whereby their adoption and deployment can occur at a greatly expedited rate.

The relative applicability of our approach is obviously dependent on the ubiquity of Trusted Platforms in the market place. This is, however, not an unreasonable assumption, and once made, allows a number of interesting solutions to a whole host of security problems plaguing CNP transactions. Currently available sales figures for 2005 [18] showed estimates of 32% of all notebook systems shipped that year being TPM-enabled. This figure is

expected to nearly triple by 2007, with processor and OS support to follow soon after. Indeed, the synergistic approach of providing a secure payment facility, through the use of e-EMV, could form a mutually beneficial arrangement for both TPM manufacturers and card issuers. Potentially leading to the faster adoption of Trusted Computing, which in turn would provide an expanded customer base for e-EMV cards.

This report is structured as follows. In Section 2, we present an overview of the system model we will use to describe our architecture. This section will also provide an outline of threats to generic Internet-based CNP transactions. In Section 3, we examine some of the proposed standards and protocols for protecting CNP transactions, as well as their various shortcomings reconciled against our threat model. Section 4 and Section 5 will look at the EMV and TCG standards respectively. These sections will act as a primer for the technicalities presented in later sections. Section 6 provides a high-level overview of our e-EMV architecture. Section 7 explains in detail the procedures and processes involved in establishing an e-EMV card within a TPM-enabled platform. Following on from this, Section 8 highlights how normal EMV transaction flows can be mapped to an e-EMV transaction. We concluded with Section 10.

2 System model

This section begins with an overview of the generic four-corner-model used in card payment systems. Following on from this, we present a high-level overview our e-EMV architecture. This section concludes with an examination of the more significant threats posed in securing CNP transactions

The model applied to card payment systems is typically referred to as the four-corner-model (or a pull-model). Within this model, a number of steps are necessary to complete a given transaction (see Fig. 1). Prior to a customer being able to interact with a merchant, it is necessary that they follow some issuer-specific enrollment procedure in order to obtain a physical payment card. A merchant, likewise, can only accept payments from a customer if they have preregistered to accept payments for that customer's particular card type with their acquirer. The dashed line represents the boundary of the financial network domain. Payment clearing occurs as follows:

- **Step 1:** The process begins with a customer signaling their intent to purchase goods by forwarding a payment record to a merchant. In this instance, the actual characteristics of a payment record differ depending on the environment in which it was created. For an on-line purchase,

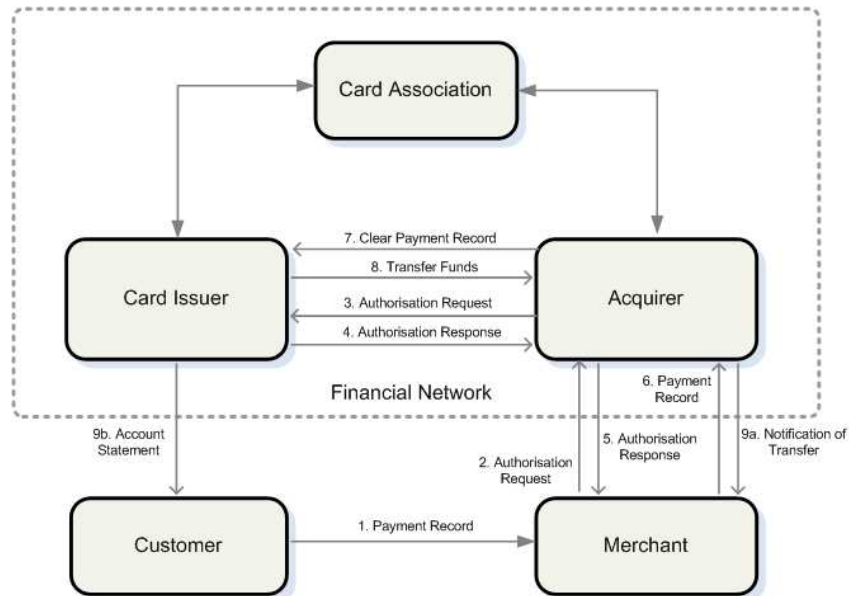


Figure 1: Generic model for card processing.

a payment record typically includes the information embossed on the customer’s physical payment card in conjunction with certain merchant supplied information (such as the invoiced amount).

- **Steps 2-5:** These steps occur immediately after receiving the customer’s payment record. They consist of a merchant submitting the transaction details to their acquirer which will either authorise or reject the transaction based on their interactions with the customer’s card issuer. After this, the merchant will either confirm payment or inform the cardholder that their transaction has been rejected.
- **Steps 6-9:** Based upon the transaction being approved, either as a result of a successful outcome from steps 2-5 or merchant risk management routines, steps 6-9 represent the account settlement process through which funds are debited from a customer’s account and credited to the merchant’s.

Perhaps the most surprising feature of this model, is that a positive transaction authorisation (step 5) does not guarantee payment for a merchant. It is merely an indication that the card account details being proffered have not been reported stolen and that the customer has sufficient funds to cover the transaction amount. Indeed, unless the card has been reported stolen,

it is impossible for a card issuer, and by extension a merchant, to ascertain whether a particular transaction is fraudulent or not.

In this regard, the merchant trusts (hopes) that the customer is the valid account holder (or at least a delegate of the primary account holder) for the presented payment record. This trust, or lack thereof, is largely underpinned by the level of indemnity offered by card issuers to their customers in the case of lost or stolen cards being used in illegitimate transactions. However, the level of indemnity afforded to merchants is dependent on their adherence to their acquirer supplied Merchant Operating Guidelines (MOG). The MOG lays out the procedures that should be followed when processing CNP transactions. An example of such a procedure would be a requirement to use an Address Verification Service (AVS) which compares the billing address, as entered by the customer, to that of the card issuer's records. If they match, this is seen as an indication that the customer owns the card being used. In many cases a merchant may be held liable for chargebacks associated with a transaction if they do not properly perform cardholder verification. This verification is more difficult to do in a CNP setting.

Our e-EMV architecture closely follows the generic four corner model as presented in Figure 1. The only differences in our approach are that the communication channel between the customer and the merchant is now exclusively Internet-based and that both card issuers and acquirers must provide enrollment facilities to their e-EMV clients, separate to the enrollment facilities for the physical issuance of an EMV card. For a card issuer this means providing a mechanism through which customer's can establish their e-EMV cards on their platforms. Similarly, for an acquirer this means providing a facility whereby a merchant can download an application that can interface with an customer's e-EMV application. In doing so, we assume the presence of a PKI, much like the one that currently exists for EMV. In this regard, enrollment facilities should be able to be authenticated by customers and merchants via public key certificates issued by a particular card association.

In presenting this architecture we are making the important assumption that TPM-enabled platforms are ubiquitous within the merchant/customer domain, and additionally that both processor and operating system support are available to all platforms. Furthermore, we make the underlying assumption that a card issuer has extended credit or debit facilities to a particular customer. For credit-based cards, prior to enrollment in this system, we assume that the applicant has been pre-approved a card using standard credit checking procedures. For debit-based cards, we assume that the applicant has an active account with the acquirer to which they are applying.

2.1 Threats

This section will attempt to enumerate the various threats faced by generic card payment systems with a specific emphasis on CNP transactions. Following on from this, in Section 3, we will examine some specific CNP-enabling protocols reconciled against the general threats presented below.

Broadly speaking the threats faced by the customer, merchant and financial network domain can be generalised into the following categories, where ordering does not reflect significance:

- **Lack of Integrity**

It is important for all parties that the integrity of the transactional data being sent is protected. Neither the merchant nor the customer should be able to modify the details of the transaction after the payment has been authorised by the customer's card issuer. We assume transaction data within the confines of the financial network domain is adequately protected.

- **Loss of Confidentiality and Privacy**

A strong positive correlation exists between confidentiality and privacy. Cardholders would like to keep their personal account details from being disclosed to unknown third parties. In this context, there is a distinction to be made between order information and payment information. Confidentiality and privacy of order information may be desirable to avoid customer profiling in which a customer's shopping habits are observed and recorded. In addition to this, some users may like to keep the browsing phase of their transactions private. This latter concern of browsing/ordering privacy isn't immediately germane to our proposal and will only receive a cursory examination in later sections. By contrast, confidentiality and privacy of payment information is essential as disclosure of account information can lead to card cloning and illegitimate use of a customer's account.

Contrary to privacy of order information, privacy of account information has received a lot of press of late. Of special note is the recent break-in at CardSystems Solutions [44] and the subsequent exposure of some 40 million debt and credit card accounts. This breach took place in spite of CardSystems Solutions passing a conformance audit for Visa and Mastercard's joint Payment Card Industry (PCI) data security standard [40]. The PCI standard is intended to provide a base set of requirements for parties involved in the storing, processing or transition of cardholder data. The obvious question arises — how do

you perform auditing to ensure compliance on an ongoing basis? This type of rolling audit would be desirable from both a customer and PCI standpoint as it would allow verification prior to processing. We will see how such an auditing system can be implemented in Section 7.

- **Lack of Authentication and Authorisation**

As attacks such as phishing and pharming become more prevalent, there is a corresponding need for customers to authenticate merchant identities prior to transaction initiation. Merchants would also like to authenticate that the customer identity claimed during a transaction is actually the one held by the transacting party. If a merchant accepts a fraudulent transaction they can be held liable for the value of the fraud as well as potentially having their processing costs increased by their acquirer. Additionally, card issuers and acquirers would like to authenticate customers' transaction data so that they can avoid authorising illegitimate payments. Additionally, card issuers would like to make sure that a valid customer with a valid card instigated and authorised each payment request. A card issuer can be liable if they authorise a fraudulent transaction.

- **Non-repudiation**

Non-repudiation is closely allied with customer authorisation. A merchant needs to have proof that the customer agreed to pay a specified amount for their goods or services. In the event of an illegitimate transaction (or possible buyer's remorse — an act whereby a customer attempts to refute a legitimate purchase), in order to avoid being held liable and paying the associated chargeback costs, a merchant would need to submit evidence to the customer's card issuer showing that the customer did indeed either, authorise the transaction, or receive the goods.

A final issue of potential concern is that of mail non-receipt. This is where a card is intercepted *en route* to a customer; this was the fourth biggest category of credit card fraud in the U.K in 2004. We will examine this in relation to the distribution of e-EMV cards in Sections 6 and 7.

3 Card payments and the Internet

In this section we will examine some of the more significant protocols and standards for protecting CNP transactions.

3.1 SSL

The Secure Socket Layer (SSL) protocol was first introduced in 1994 by the Netscape corporation. The protocol itself was designed to provide end-to-end security services to connections running over TCP/IP and has since become the de facto standard for the secure transmission of CNP transaction information. However, this use of SSL can be seen as more of a highjacking of an existing technology rather than a systematic approach to securing CNP transactions. In this regard, SSL establishes a session between a customer and a merchant and acts as a facilitator for the secure transfer of account details, of which, quintessentially, the PAN, CSC and relevant billing information are all requisite elements. SSL's primary advantage, and perhaps the main reason for its pervasive deployment, is that it requires no additional equipment for a cardholder and not much additional inconvenience for a merchant. However, what happens outside of an established transfer session is not within the scope of SSL's protection remit.

In this respect, the confidentiality and integrity afforded by SSL only protects against attacks from parties attempting to eavesdrop on a transaction between a customer and a merchant. It says nothing as to validity of the data emanating from either end-point. Potentially the biggest deficiency in the use of SSL for CNP payments is the lack of customer authentication. Even though SSL provides a provision for client (customer) authentication, it is seldom, if ever used. This stems from the inconvenience and cost associated with distributing and managing client certificates. A further issue relevant to client certificates, as mentioned in Section 1, is the problem of the perpetual increase in malware-affected platforms. If the private component of a key bound to a client SSL certificate is exposed to malicious software on a platform, then it becomes impossible to attest with any certainty that an entity purporting to be certified is as claimed.

3.2 SET

SET differs from SSL in that it was designed explicitly as a payment protocol and addresses a number of the deficiencies found in the SSL-based approach for facilitating on-line card-based commerce. However, despite improvements over an SSL-based approach, SET is no longer being deployed for use in CNP transactions. A number of theories have been put forward to explain why SET never became a success. These range from ease of use to the cost and difficulty of maintaining a stable PKI. For a more through treatment of SET than the one presented here, we refer readers to [36, p.100-123].

SET allowed every entity that was party to a transaction to be authenti-

cated. SET used a certification authority hierarchy in which all participants were required to enroll. Certificates were then exchanged allowing authentication to occur. When it came to making a purchase within SET, a purchase order message would be constructed in such a way that only the merchant could see the Order Information (OI) and only the payment gateway could see the Payment Information (PI). This was accomplished through what was termed a ‘dual signature’, whereby messages intended for the merchant and messages intended for the payment gateway could be linked without simultaneously revealing both. In this instance, the PI comprised transaction related data as well as a transaction ID which were then carried in a “digital envelope”. A one time session key was created for bulk encryption which then encrypted the PI, the dual signature and a hash of the OI. This key was then enciphered with the public key of the payment gateway. Only the gateway could decrypt the said envelope and obtain the key to decrypt the enciphered PI.

Authorisation was more complex insofar as a payment needed to be authorised by both the customer and their bank. The merchant forwarded the encrypted PI information to the payment gateway along with their own authorisation information. This information comprised, amongst other things, an authorisation block containing a transaction ID encrypted with a session key and signed with the merchant’s private key. This allowed the payment gateway to verify both parties by their respective signatures as well as confirming that they were referring to the same transaction by comparing ID values. One of the nice features of SET was that it did allow for non-repudiation of transactions — assuming the transaction was authorised by the card issuer.

3.3 3-D Secure, Verified by Visa and SecureCode

3-D Secure and both Visa’s [5, 6] and MasterCard’s [31] proposals, Verified by Visa (VbV) and SecureCode respectively, attempt to provide cardholder authorisation for Internet-based CNP transactions, and in this respect, can be seen as an adjunct to the SSL-based approach outlined in Section 3.1. Both proposals are designed solely to provide cardholder authorisation and both require customers to preregister their account with their card issuer prior to using the system. During the registration procedure the cardholder chooses a secret password that will later be used to authorise subsequent CNP transactions. These authorisations may later act as non-repudiable evidence in case of a dispute. Both the VbV and SecureCode proposals provide equivalent functionality (as they are both derivations of 3-D Secure), so we will concentrate our discussion on Visa’s proposal as an illustrative

example.

In the VbV approach, during the payment phase of a transaction a customer's browser is redirected by a merchant plug-in component to an appropriate ACS for their account. The customer authenticates to this ACS by providing their username and password, as established in the registration phase. Based upon the correctness of the supplied username/password combination, the ACS formulates its response (authenticated/not authenticated) and signs it. This signature is then passed through the customer's browser and onto the merchant plug-in. The plug-in then verifies the ACS signature and decides if it wishes to proceed with the transaction. A validated response can later be used as evidence to show the customer authorised a particular payment. If the customer account number is not registered with any ACS, a visa directory server informs the merchant plug-in and normal MOTO-based authorisation procedures are attempted.

The use of 3-D secure (and its derivatives) can be seen as forcing an additional customer authentication prior to the completion of a transaction. However, given the nature of current implementations, especially with regard to the static nature of current authentication information (based on passwords), it is difficult to see how authoritative this authentication would be, and how non-repudiable the evidence of transaction authorisation would be. Indeed, there are various threats that affect the security of any CNP proposal, most notably spyware and phishing attacks. However, 3-D Secure's real benefit comes in reducing the economies of scale possible with card skimming attacks. An attacker obtaining a customer's card details, possibly by means of a compromised POS terminal, will no longer be able to complete a fraudulent purchase using the obtained information as a PAN and CSC are no longer sufficient to authorise a CNP transaction authorisation. Unfortunately, in this instance the use of a static authenticator may prove no less of a barrier to obtaining card account details. Perhaps the greatest threat to such a scheme would be that of an automated attack script that compromises cardholder platforms and installs malware that monitors keyboard activity and generate new transactions using the observed authorisation data. Additionally, a phishing site that purports to provide a 3-D secure plug-in capability could potentially dupe cardholders into revealing authentication data.

3.4 Summary

Of the three protocol suites presented thus far only SET is no longer being deployed. The SETCo web-site¹ is inoperative and the standard is conspicuous by its absence from the latest version of the EMV specifications [13]³.

There are a number of theories put forward as to why SET never became a success ranging from ease of use to the cost and difficulty of maintaining a stable PKI. However, with regards to the threats posed in Section 2.1 SET fared rather favorably in comparison to SSL. Correspondingly, the 3-D Secure proposals, with the possible exception of customer authentication and non-repudiation, has similar security defects as an SSL-based approach to securing CNP transactions. In fact 3-D Secure relies quite heavily on SSL/TLS to secure the communications between each entity involved in a transaction. In both 3-D Secure and SSL customer account information is sent to the merchant with no guarantees as to privacy or confidentiality of this information. Of the three proposals, SSL is by far the most ubiquitous in its deployment. However, SSL suffers quite badly from its lack of client-side (customer) authentication.

4 EMV

In the United Kingdom the use of magnetic stripe cards for point of sale transactions has been phased out in favour of EMV-compliant Integrated Chip Cards (ICCs). The susceptibility of magnetic strip cards to cloning and the projected fraud losses associated with the continued use of this technology, estimated to be in the region of £1 billion per annum by the end of the decade in the U.K [3], has resulted in a migration to the more resilient EMV ICCs. In this instance the cost of migrating from magnetic stripe cards to EMV ICCs (which was less than future projected losses from fraud) catalyzed the transition process. France, which has been using a chip-and-pin based system for the last 12 years, saw a post-migration reduction, on the order of 80%, in fraud on domestic transactions [33].

This section aims to provide a brief overview of the EMV specifications, particularly in relation to transaction processing and EMV key usage. EMV, as described in the current iteration of the standard [11, 12, 13, 14] defines the full spectrum of interactions, from the physical to the logical, between

¹<http://www.setco.org/>

³There had previously been an annex in version 4.0 of the specification [10] outlining the integration of SET and EMV for enabling “transaction processing for chip electronic commerce”.

an ICC and an Integrated Chip (IC) enabled POS terminal. EMV supports both cardholder authentication through new Cardholder Verification Methods (CVMs) and ICC authentication through either Static Data Authentication (SDA), Dynamic Data Authentication (DDA) or Combined DDA and application cryptogram generation (CDA). The introduction of new CVMs and SDA/DAA/CDA functionality aims to reduce credit and debit card fraud across the board through improved authentication and authorisation mechanisms. Additionally, depending on the results of card and terminal risk management routines, which are designed to protect issuers and acquirers respectively, transactions can be completed either on-line or off-line.

4.1 Transaction processing

The processing of an EMV transaction begins when the card is inserted into a POS terminal and comprises of a number of distinct stages. However, the stages, as presented here, do not follow a linear path. Some of them operate under a not before but not after semantic and some of the steps, where indicated, are optional.

1. **Application Selection:** The terminal issues a SELECT command [11, p.129] to choose the appropriate EMV application from a multi-application card. The selection procedure can be either direct or indirect and is based on the same Application Identifiers (AID) being supported by both the terminal and the ICC.
2. **Initiate Application Processing:** The terminal commences application processing by issuing a GET PROCESSING OPTIONS command [13, p.63] to the card. In response to this the card returns the Application File Locator (AFL) and the Application Interchange Profile (AIP). The AFL prides a table of contents like structure of the publicly available information provided by the card application. The AIP indicates the application functions that are supported by the card such as if off-line SDA/DDA (Static Data Authentication/Dynamic Data Authentication) is available or if card holder verification is supported.
3. **Read Application Data:** A number of READ RECORD commands [13, p.69] are issued by the POS terminal to the card to read the necessary data for the transaction to proceed. As part of this step, the terminal makes sure all mandatory data elements are present on the card.

4. **Processing Restrictions:** This is a mandatory step performed by the terminal and doesn't require any interaction with the ICC. The purpose of this step is to ascertain the level of comparability between the application running on the card and the application running on the terminal [13, p.100]. Based on this comparison, the terminal takes an appropriate action — including possibly terminating the transaction.
5. **Off-line Data Authentication:** This is an optional step in which the terminal issues an INTERNAL AUTHENTICATE command [13, p.65]. There are three options for off-line data authentication available in EMV-compliant cards. The first is SDA, the second is DDA and the third is CDA. SDA allows the terminal to determine if the personalised data on the card has been modified since personalisation. DDA is more complex in that it allows for the same checks as SDA but also enables the terminal to determine if the card is genuine or not through a challenge response mechanism. This involves the ICC generating a dynamic signature (see Section 4.2.1) over the terminal supplied challenge. CDA is similar to DDA except the dynamic signature generated by the ICC includes an Application Cryptogram (AC) (see Section 4.2.2) which can be verified by the terminal to provide a degree of non-repudiation.
6. **Card holder Verification:** This optional step [13, p.103] allows the terminal to potentially verify the authenticity of the cardholder in relation to the card based on a particular CVM. It is also used by the cardholder as a means to authorise a particular transaction. CVM methods range from no verification required to both off-line and on-line PIN verification.
7. **Terminal Risk Management and Action Analysis:** The purpose of terminal risk management [13, p.107] and terminal action analysis [13, p.111] is to protect the issuer, acquirer and payment system from fraud. Based on variety of measures, such as floor limits, random transaction selection or velocity checking a transaction may be forced to complete on-line.
8. **Card Risk Management and Action Analysis:** Details of card risk management are proprietary to card issuers and are outside the scope of the EMV specification. As a result of card action analysis [13, p.115] (which is based on the results of the card risk management routines), an ICC may decide to complete a transaction by either going

on-line, remaining off-line, requesting a referral or rejecting the transaction outright.

9. **On-line Authorisation:** During normal transaction processing the terminal may decide to proceed with an on-line check. This could be as a result of a number of factors. The terminal may only support on-line transaction processing or the store attendant may have found the customer suspicious. It could also be as a result of the outcome of either Steps 7 and 8. In any case, a transaction was deemed outside the risk profile for off-line completion and requires further scrutiny by the customer's card issuer. An AC is generated by the card and forwarded to the card issuer which, after examination, determines if the transaction should go ahead or not. The card issuer returns another AC informing the card whether or not to proceed with the current transaction.
10. **Issuer-to-Card Script Processing:** In order to allow card updates in the field, the card issuer can return scripts via the terminal to the ICC for processing. These scripts are not necessarily relevant to the current transaction and may be used to update applications during the utilisation phase of the ICC's lifecycle, or to transition the card into a blocked or unblocked state.

4.2 EMV Key Usage

4.2.1 SDA and DDA

The number of keys in an EMV-compliant ICC depends on whether the card is SDA or DDA compliant for off-line authentication. In the case of SDA, the ICC maintains the public key of the issuer (in a certificate signed by a card association CA) as well as a signature of its static application data created using the card issuer's private key. The terminal, in turn, stores the public key of the card association CA in order to verify the signed static application data (by obtaining and verifying the card issuer's public key from the ICC).

DDA is similar except that in addition to the SDA keys, the card itself has its own key pair, of which the public part is stored in an ICC certificate. This certificate is signed with the private key of the card issuer. Retrieval of the public key by the terminal is identical to SDA except for one additional step. The terminal uses its embedded CA public key to verify the certificate containing the issuer's public key which it then uses to verify the ICC certificate. If this verification is successful the terminal extracts the ICC public key from the ICC certificate. In DDA, the ICC public key is used

to generate a signature over the static application data as well as a random number provided by the terminal. This allows the terminal to determine whether the card is genuine or not as well as being able to ascertain if the data personalized in the card has been altered since card personalization, as per SDA.

4.2.2 Application Cryptograms

Application Cryptograms are generated by the ICC and card issuers using MACs based on a shared secret. The ICC and its issuer share a long-term MAC master key, the ICC Application Cryptogram Master Key MK_{AC} . Application Cryptogram Session Keys, SK_{ACs} , are derived from these master keys using a session key derivation function (as defined in Annex A1.3 of [12]) that uses a card's Application Transaction Counter (ATC) as diversification data. An ATC is a counter that keeps track of the total number of transactions performed by a card. In this way an SK_{AC} will be different for every AC generation.

To avoid the issuer having to store, a unique master key for every card the master key is itself a result of a derivation process. The issuer uses the ICC's Primary Account Number (PAN), the PAN sequence number as well as its Issuer Master Key (IMK) to dynamically generate the shared MAC master keys.

In the EMV specifications session keys are used to protect transaction messages. EMV defines four different types of transaction messages (ACs). Depending on the reconciliation of card and terminal risk management routines one of the following will be generated by the card: an Application Authentication Cryptogram (AAC), an Application Authorisation Referral (AAR), an Authorisation Request Cryptogram (ARQC) or a Transaction Certificate (TC). If the transaction is approved, the ICC will generate a TC which will be passed to the terminal and can be used to claim payment during the clearing process. If a transaction is declined, the ICC will generate an AAC. In the event that the transaction needs to be approved on-line the ICC generates an ARQC which will be forwarded to the issuer. The issuer will then reply with an ARPC indicating whether the transaction should be approved or declined, in which case a TC or an AAC will be generated by the ICC.

4.2.3 Cardholder Verification

The EMV standard specifies a number of CVMs ranging from: no CVM required, to signature-based CVM, to both on-line and off-line PIN verification.

In supporting off-line PIN verification, the card maintains a public key pair for PIN encipherment. Typically, this PIN encipherment key pair will be the same as the ICC key pair. The terminal may use the public portion of this key pair (depending on terminal's support for either plaintext or enciphered PIN verification) to encipher the PIN entered from the PIN pad which the terminal forwards to the ICC for verification. Based on the outcome of this verification the ICC determines if it wishes to proceed with the transaction or not based on card action analysis routines.

5 Trusted Computing

This section, in conjunction with Section 4, is intended as primer for Sections 6 and 7 by describing and outlining the trusted building blocks on which we build our e-EMV architecture. Trusted Computing as discussed here, relates directly to the type of system espoused by the Trusted Computing Group (TCG). In this case, a trusted system is one that behaves in the expected manner for a particular purpose.

The current documentation from the TCG is a rather nebulous set of specifications encompassing TPM design [24, 25, 26], specifications for trusted networking [21, 22, 23] as well as PC [17] and server specifications [27]. This, however, is by no means the totality of effort invested in Trusted Computing. Trusted Computing also encompasses processor design [30, 2] as well as Operating System support [34, 37]. We will now examine the role TPM, OS and processor play in the design of trusted systems.

5.1 TPM Specification

The TPM forms the core of all efforts in Trusted Computing. The TPM itself comes in the form of a microcontroller with Cryptographic Coprocessor (CCP) capabilities that resides on a platform's motherboard. The TPM, in conjunction with offering a secure storage area for keying material, is capable of providing the following functionality:

5.1.1 Protected capabilities and shielded locations

The TPM provides a number of internally shielded locations that are only accessible through the use of protected capabilities. These locations provide secure areas in which the platform can operate on sensitive data. Perhaps the most important shielded locations in terms of any discussion of Trusted Computing functionality are Platform Configuration Registers (PCRs). A

PCR is 20 byte storage area that acts as a repository for recording integrity altering events within a platform.

5.1.2 Integrity Measurement and Storage

Trusted Computing defines several roots of trust within a TPM. Integrity measurement and storage fall under the purview of the Root of Trust for Measurement (RTM) and the Root of Trust for Storage (RTS) respectively.

The RTM is a computing engine that is responsible for making intrinsically reliable integrity measurements. By the same token, the RTS is responsible for maintaining an accurate and sufficiently detailed record of the events measured by the RTM. The use of integrity in this context refers to ensuring that state transitions are accurately reflected in the current platform state. This differs slightly to the type of integrity espoused by EMV which concerns itself with card details not being modified since personalisation. However, as we shall see in Section 7, this static EMV-based definition can be modeled by the more dynamic Trusted Computing based definition. Indeed we show how, in modeling this EMV definition, how we can retain the dynamism offered by Trusted Computing whilst remaining true to post-personalisation context offered by EMV.

1. **Measurement:** Measuring events within a platform is a two-stage process that begins with an *extend* command. This command, more commonly referred to as ‘extending the digest’, appends a hash of the event being measured to one of a number of PCRs located internally to the TPM. These PCRs store a representative hash of all the events generated by a platform, forming a picture of the current platform state. This digest extension process can be illustrated as follows:

$$PCR_x = HASH(PCR_x \parallel measured\ value)$$

Here PCR_x is the digest being updated. The use of this approach guarantees that related measured values will not be discarded as the previous register value is incorporated into the new register value. The use of *measured value* is a representation of embedded data or program code executing within a platform.

The other stage in this process is the writing of events, as reflected in a particular PCR, to permanent storage. This writing to storage, or logging, of integrity altering events within a platform occurs in the Stored Measurement Log (SML). The SML maintains sequences of events to

which new events are appended. Due to storage constraints this log will typically be located externally to the TPM.

The two most important properties abstracted from the brief discussion outlined above are firstly, that atomicity of extensions be ensured, and secondly, that updates be noncommutative. That is to say, updates should follow an all or nothing semantic and the order in which extension events occur matters.

2. **Storage:** Storage, as defined by the TCG refers to two distinct concepts. The first, which we just covered, refers to the intermediate step between measurement and reporting. The second, which we will discuss in the next section, refers to the protection of keys and data within a TPM. The responsibility for the protection of said keys and other sensitive data within a TPM falls under the oversight of the RTS. In this regard, the RTS maintains a small area of volatile memory used for performing cryptographic operations and storing data.

5.1.3 TPM Keys and Key Usage

TPM Keys are stored in a tree structure with the Storage Root Key (SRK) forming the root of all keys stored within the TPM_Key tree hierarchy. Within this hierarchy every node (TPM_Key) has an assigned attribute designation as well as a defined key type. Attribute designations, in the context of a TPM_Key, help to define key mobility. A key can be designated either migratable, certified-migratable or non-migratable. Keys with either of first two designations are capable of leaving a TPM, whilst non-migratable keys are inextricably bound to a single platform. Key types are used to define what particular operations a TPM_Key is capable of performing. For example, a key could have an signing type or storage type depending on the designation of its usage.

Due to the limitations in available memory, it is necessary to move inactive keys into off-chip storage areas from time to time. This process is managed by the Key Cache Manager (KCS). Inactive keys are first encrypted into a “blob” using a key controlled by the SRK. These blobs are opaque outside of the TPM and may be bound to the platform on which the TPM resides. Binding, in this context means that blobs may only be decrypted on the platform that created them, as the decryption key may be fixed to a particular TPM. The exception to this rule is the case where the wrapping key responsible for creating the blob is migratable.

In addition to the attribute designations and key types outlined above, there are two key pairs in particular that hold special significance in Trusted

Computing. These key pairs are the Endorsement Key pair (EK) and the Attestation Identity Key (AIK) pair. Within a TCG-conformant platform, AIK key pairs act as aliases for the EK and are responsible for attesting platform states. AIK pairs are used because an EK pair is unique per TPM instance and this is considered a possible risk to user privacy should the EK pair become connected with personally identifiable information. As there is no prescribed limit on the number of AIKs that can be used within a platform, this provides an anonymity mechanism, whereby the TPM can use different AIKs each time it attests to platform integrity metrics.

5.1.4 Reporting and Attestation

The final root of trust within a Trusted Platform is the Root of Trust for Reporting (RTR). Its function is to faithfully recount information held by the RTS to third parties. The mechanism through which this is achieved is referred to as ‘attestation’.

Attestation within the context of Trusted Computing is the mechanism whereby a TPM-enabled platform adduces certain state characteristics that govern a platform’s trustworthiness. This evidence is consumed by integrity verifiers that use this evidence, in conjunction with the perceived trustworthiness of the entity performing the attestation, to establish a trust level in the attesting platform.

The basic steps involved in an attestation are as follows:

1. An external entity requests one or more PCR values.
2. A platform agent culls the SML for the events responsible for generating the requested PCR values.
3. The TPM signs the requested registers using an AIK by calling the TPM_Quote command [26, pp.155].
4. The agent then obtains various credentials (the platform credential, the conformance credential, attestation identity credential — see Section 5.1.6) that vouch for the TPM. These credentials, along with the relevant portion of the SML and the requested signed PCR values are returned to the requesting entity.
5. The requesting entity then examines the credentials, checks signatures and compares a hash of the received SML entries to the attested PCR values. If they match the verifier can gain some assurances as to the current state of the attestor’s platform at the time the attestation was made.

How this attestation is actually performed differs between version 1.1b and version 1.2 of the specifications. Version 1.1b uses what is referred to as the “Privacy CA” model whilst version 1.2 introduced a new model in the form of Direct Anonymous Attestation (DAA).

Privacy CA

The Privacy CA approach is employed as a means of issuing AIK credentials (see Section 5.1.6) to TPM-enabled platforms. These credentials are used to vouch for the trustworthiness of an attestation originating from a platform. credential acquirement is achieved as follows: a `Tspi_TPM_CollateIdentityRequest` command [28, pp.111] is issued by a platform prior to the generation of an AIK key pair, this command gathers all the required information necessary for a Privacy CA to examine the requestor’s platform. This information includes various credentials that vouch for the trustworthiness of the TPM itself. Provided the evidence presented by a user’s platform is validated by the Privacy CA, the Privacy CA will encrypt the newly generated AIK credential with a symmetric key, which in turn is encrypted with the EK of the requesting platform. In this way only a specific platform is capable of decrypting the credential and performing the `TPM_ActivateIdentity` command [26, pp.151]. This then allows an AIK private component to be used to generate signatures over platform integrity metrics.

There were, however, significant concerns raised over the ‘privacy’ afforded to this solution. In this model, the Privacy CA is capable of linking individual AIK public keys to a particular TPM. This was due to the fact that in order to obtain an AIK credential it was necessary for a TPM to disclose its AIK — EK binding. Potentially leading to the exposure of personally identifiable information in the process.

DAA

Direct Anonymous Attestation (DAA) by contrast is a group signature scheme without an anonymity revocation property. DAA was designed as an optional extra for version 1.2 of the TPM specification and aims to address some of the privacy concerns that arose in the Privacy CA model. DAA differs from the Privacy CA model in that it doesn’t disclose AIK keys to an issuer, and hence aims to limit the exposure AIK — EK bindings.

Enrolling a TPM-enabled platform with a DAA issuer occurs when a platform instigates a run of the DAA join protocol. During this protocol

the platform authenticates itself to an issuer via the public component of its EK and proves (in the form of a zero-knowledge proof) that it knows a non-migratable, TPM controlled secret value f . In addition to this, the platform provides a pseudonym $N_I = \zeta^{f_P}$, where ζ is derived from the issuer's long term base name and f_P is a secret associated with the TPM-enabled platform.

Providing the issuer finds the above acceptable, it will issue the platform a credential in the form of a Camenisch-Lysyanskaya (CL) [9] signature on the TPM-controlled secret value f . This signature will later be used to convince a verifier that it has previously completed a run of the DAA protocol with respect to a particular issuer. In this way a DAA issuer never becomes aware of the relationship between an individual platform, as identified by its EK pair, and its AIK pairs.

Subsequent to a run of the DDA join protocol, the DAA-Signing protocol can be used to generate a DAA signature on a AIK public key. During this process, the platform is only identified by a pseudonym. As in the join protocol, this pseudonym is of the form $N_V = \zeta^f$, where now the selection of ζ determines the anonymity properties of the attestation process. The value of the base ζ will typically be chosen by the verifier.

5.1.5 Delegation and Migration

Delegation and migration are new features that were added to version 1.2 of the TPM specifications. The delegation model used in TPM v1.2 aims to address some of the concerns surrounding the 'super-user' like privileges afforded to TPM_Owner. The concept of a TPM_Owner refers to an entity that inserted a shared secret into a TPM shielded-location during the TPM enablement process. Knowledge of this secret confers the right to execute certain protected capabilities within a TPM. Previous to version 1.2 of the TPM specifications the delegation model that was employed required the revelation of the TPM_Owner secret in order to allow a subordinate process access to a particular protected capability. In order to better apply 'the principle of least privilege' a new model was introduced. This new model involves the creation of new authorisation data and inferring certain owner privileges to it. In this way delegation can be much finer grained as it does not require the revelation of the TPM_Owner's secret.

Migration is also a new concept introduced in the TPM v1.2 specifications. Migration is an operation that permits the secure transfer of migratory keys from one TCG-compliant platform to another. This transfer occurs in

such a manner as to allow the new platform full usage of the migrated key. The notion of a migratable key can be subdivided into two categories, a Migratable Key (MK) and a Certified Migratable Key (CMK). MKs are keys that, while not bound to a specific TPM, can be transferred to another TPM if appropriate authorisation is provided. CMKs are similar, insofar as they also require appropriate authorisation in order to migrate. However, their migration is conditional on receiving permission from a relevant Migration Authority (MA). For more details pertaining to the use and operation of CMKs we refer the reader to Appendix A.

5.1.6 TCG Credentials

The issue of certification plays an important role in defining what is meant by a “Trusted Platform”. In this regard there are five interrelated credentials that adduce the existence of a correctly functioning TPM within a platform.

- **Endorsement credential:** This credential vouches that a TPM is genuine and is issued by whomever generated the EK pair. Typically this will be the TPM manufacturer or vendor.
- **Conformance credentials:** These credentials vouch that the TPM design conforms with established evaluation guidelines as laid out by the TCG. These credentials are issued by entities with sufficient credibility to evaluate a TPM. Typically a conformance testing facility.
- **Platform credential:** This credential provides evidence that a platform contains a TPM and incorporates references to both the endorsement and conformance credentials. This is typically generated by a platform vendor.
- **AIK credentials:** These credentials provide evidence as to the existence of an AIK within a TPM. These credentials incorporate references to endorsement, conformance and platform credentials and are typically generated by a Privacy CA.
- **Validation credential:** This credential provides signed reference measurements (digests of the measured components taken during development when it is believed to be in a stable condition) which can be compared against runtime measurements to assure interested parties that their application is performing as intended. These runtime measurements will be signed by AIK public keys.

5.2 Trusted PC and Trusted Server Specifications

Both the Trusted PC [17] and Trusted Server [27] specifications provide generic implementation reference documentation for building Trusted Platforms at the client and server side respectively.

The Trusted PC specification deals primarily with instantiating the various elements involved in transitioning a platform from its pre-boot to its post-boot state. Within this transition, various usage constraints are defined for TPM components, such as usage of PCR registers as well as guidelines for option ROMs. The specification also outlines programmatic interfaces to the BIOS in the form of the TCG Software Stack (TSS) as well as detailing how physical presence is demonstrated on a TPM enabled platform.

The Server specification details much of the same functionality as the Trusted PC specification. However, it introduces a number of new concepts that are of special interest in server architectures. In particular it highlights the distinctions between partitions and platforms. The specification defines a partition as: “the hardware and firmware environment, which provides the support for the execution of a single operating system image within a separated trust environment”. A platform is later defined as: “the complete package of physical hardware and firmware that a manufacturer produces for a single server product. This may be configured as one or more partitions.” The issue of dynamic reconfiguration is addressed insofar as stating the measurement of asynchronous events are the responsibility of the post-boot software environment.

5.3 Trusted Server Specification

5.4 TNC Specification

The Trusted Network connect (TNC) specification forms an expatiated subclass of the Infrastructure Work Group (IWG) interoperability specification [20] and deals predominantly with enabling the enforcement of operator controlled policies for endpoint security in determining network access.

TNC can be seen as an enhancement to the IETF’s AAA authorization frameworks [41, 42, 43] in offering a way of assaying an endpoint’s integrity to ensure it complies with a particular predefined policy. A particular instance of this would be ensuring that a certain software state exists on a platform prior to the platform being granted network access, for example, requiring anti-viral or software patch updates to be installed. The means through which this is achieved follows a three phase approach of assess, isolate and remediate which we briefly discuss next.

The assess phase deals predominantly with an Access Requestor (AR) wishing to gain access to a restricted network. In this phase the Integrity Measurement Verifier (IMV) on a Policy Decision Point (PDP) examines the integrity metrics coming from the Integrity Measurement Verifier (IMC) on the AR's platform and compares them to its network access policies. From this process of reconciliation the PDP informs a Policy Enforcement Point (PEP) of its decision pertaining to an AR's access request. The PEP is then responsible for enforcing the PDP's decision. As an extension to the assessment phase, in the event that the AR has been authenticated but failed the IMV's integrity-verification procedure, a process of isolation may be instigated whereby the PDP passes instructions to the PEP which are then passed to the AR directing it to an isolation network. The final phase, remediation, is where the AR on the isolation network obtains the requisite integrity-related updates that will allow it to satisfy the PDP's access policy.

5.5 OS and Processor Support

Both Operating System and Processor support represent integral components in the realisation of Trusted Platforms. The process begins with a TPM providing a secure initialisation facility which measures the operational state of an OS as a platform transitions from a pre-boot into a post-boot state. If this post-boot state matches some agreed upon "good" value then the OS can be seen to be functioning correctly. This correctly functioning OS can then provide a stable baseline from which the future execution of programs can be measured. As well as providing access to TPM functionality, a Trusted OS will provide sandboxed execution environments in which applications can run. In this respect, processor and chipset extensions will provide the hardware support for the creation of these protected environments and act as a basis for enforcing application level sandboxing within main memory.

In this regard, Microsoft's Next Generation Secure Computing Base (NGSCB) [34, 37] forms an illustrative example of OS level security services for a Trusted Platform. The following are generic security services for a Trusted OS based on an isolation kernel and we assume the presence of such functionality in our later discussions.

1. No interference: Ensuring that the program is free from interference from entities outside its execution space.
2. Trusted path: Assumes a trusted path between a program and an input device.

3. Secure inter-process communication: Enabling one program to communicate with another, without compromising the confidentiality and integrity of its own memory locations.
4. Non-observation: An executing process and the memory locations it is working upon should be free from observation.

With regard to processor support. There have been a number initiatives in this domain, namely Intel's LaGrande, AMD's Pacifica and ARM's TrustZone. Ostensibly, each technology provide similar functionality. However, TrustZone is targeted at the embedded systems market whilst LaGrande and Pacifica target PC client and Server architectures. All three technologies aim to provide hardware enhancements that enable compartmentalised memory locations for applications to execute in. In this respect a Trusted processor provides the underlying service on which an OS can build its services.

6 An overview of e-EMV

This section aims to present a high-level overview of e-EMV. In doing so we introduce a number of topics that are essential to the establishment of an e-EMV card within a Trusted Platform. The aim here is to provide functionality akin to that of a standard EMV card by replicating EMV functionality through procedures and capabilities natively supported by a TPM-enabled host. Allowing us to provide a secure and extensible architecture for the enablement of CNP transactions.

The procedures for establishing an e-EMV card on a TPM-enabled platform comprise of a two-stage process consisting of an account activation, and an application delivery stage. The process of emulating standard SDA/DDA/CDA compliant cards necessitates a key generation procedure as the requisite keys need to be generated local to a particular TPM. Keys generated external to a TPM fail to fall directly under the purview of the protection mechanisms provided by a Trusted Platform.

In describing this architecture we are making the following important assumptions, as reiterated from Section 2. We assume the presence of a PKI, much like the one the currently exists for EMV. That TPM-enabled platforms are ubiquitous within the merchant/customer domain, and additionally that both processor and operating system support are available to all platforms within this domain. Furthermore, we make the underlying assumption that an card issuer has extended credit or debit facilities to a particular customer.

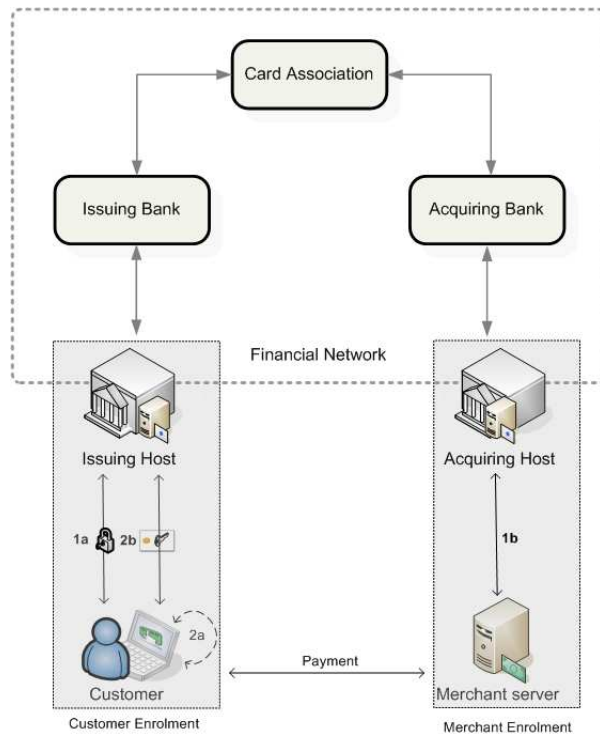


Figure 2: Architecture for Enrollment.

6.1 Enrollment

Enrollment in the e-EMV architecture is the act whereby a customer formally registers as a legitimate cardholder and obtains an e-EMV card within the system. Much like enrollment in the traditional EMV architecture, a card issuer within the system is responsible for enrolling cardholders as well as later authenticating their transactions (and possibly the cardholders themselves) via their supplied credentials. Acquirers can be seen to provide similar functionality to their merchant customers.

The procedures for establishing an e-EMV card on a TPM-enabled platform are as follows:

1. Account Activation:

The first stage, or the account activation stage, is the process through which a customer becomes a member of a card issuer controlled group. In this case, group membership is indicative of a customer activating their account within the system (Figure 2, Step 1a). In the context of Trusted Computing, this means becoming a member of a DAA (Di-

rect Anonymous Attestation) or Privacy CA group (see Section 5.1.4). This is achieved by demonstrating the presence of some non-migratable TPM-controlled secret over which certification is requested, either an AIK in the case of the Privacy CA or a secret value f for DAA.

At this point, the actual mapping between the customer and their platform can be accomplished through a mechanism of the issuer's choosing. Such as, the customer providing information supplied to them in the pre-enrolment stage (in which the card issuer agreed to extend debit/credit facilities) communicated over a secure channel. In addition to this authentication information a platform will send various credentials (see Section 5.1.6) as well as evidence as to the existence of a non-migratable TPM-controlled secret. After reception of this information, the card issuer performs due diligence in satisfying itself as to the relationship between a customer and their platform. This is achieved through a reconciliation of the provided authentication information with an examination of evidence supporting the existence of a TPM-controlled secret. If the card issuer is satisfied as to this binding, the customer's platform will receive certification on their TPM-controlled secret. This certification will later be used to demonstrate membership of a particular card scheme.

2. Secure Application Delivery:

Sometime after the first stage is complete, the customer downloads a small secure application bundle (See Section 7.2) that provides acts as a guide through the process of creating/installing the requisite TPM managed keys. This bundle, once installed will enable a platform to perform electronic transactions (Figure 2, Steps 2a and 2b). In addition to interfacing with the TPM to provide a key-generation facility for the creation of AC keys, the application itself has the dual role of acting as a user-interface, fulfilling the application requirements of a typical EMV card-resident application. The application will also provide some of the functionality typically seen in POS terminals, particularly when it comes to cardholder verification, as we shall see Section 8. At this point it is important to note that the program used to guide the customer through this process (in both stages 2a and 2b) is not necessarily a stand-alone piece of software: it could very well be an applet or browser plug-in.

Enrollment for the merchant (Figure 2, Step 1b), by contrast, is a function of satisfying the requirements for payment processing as laid down by their acquirer's MOG. During the merchant enrollment procedure the merchant's

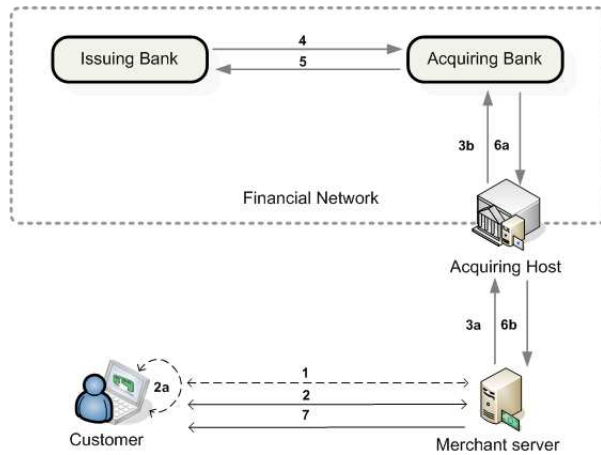


Figure 3: Architecture for TPM assisted EMV payments.

acquirer also becomes a certificate issuer, in the context of the Privacy CA or DAA models. Here the merchant will need to install a plug-in, similar in applicability to a 3-D Secure Merchant plug-in [6] but capable of emulating certain EMV terminal functionality. The most important of which will be the authentication of customer supplied credentials. Much like terminals in the physical setting, merchants will require card issuer’s public key certificates in order to verify customer transactions.

6.2 Transaction Architecture

After the enrollment procedure is complete, the e-EMV cardholder is now ready to purchase goods at any participating TPM-enabled merchant. The semantics of trust enforced by Trusted Computing functionality enables both parties, the merchant and the customer, to obtain certain guarantees that were hitherto unrealizable in past proposals. They can now be sure as to the integrity of their communication peer’s platform. That each peer will behave in the expected manner, in this case, adhere to, and faithfully adduce state characteristics corresponding to legitimate transaction states. It is important to point out that we do not expect the customer to be able to recognise or validate software states within the system. This function can be fulfilled by the application software itself and simply reported to the customer in a way that is understandable.

The basic flow for transaction processing follows the message flows presented in Figure 3:

- Step 1 - Represents the browsing phase in which a customer browses a

particular merchant site. During this step, the customer verifies that the merchant is a member of a valid group of merchants corresponding to a particular DAA/Privacy CA issuer (acquirer). Additionally, the customer may verify that the merchant is in a state that exemplifies an adequate policy for addressing privacy and confidentiality concerns. For example, conformance with Visa and Mastercard's joint Payment Card Industry [40] (PCI) data security standard.

- Step 2 - Represents the typical EMV ICC terminal interaction except that the communication channel is now over the Internet.
- Step 2a - Represents the creation of an AC. As part of this phase the customer's platform generates a signature over its AC (using the non-migratable TPM-controlled secret established during the account activation phase) as well as the PCR values that provide evidence as to the platform's current state at the time of transaction authorisation.
- Steps 3 to 7 - Represents the decision to go on-line. As a result of either terminal or card risk management procedures, the application cryptogram (possibly in conjunction with the information adducing the current platform state) is forwarded to the customer's card issuer. After examining the received data, the card issuer returns an AC of its own. This AC informs both the card and the merchant as to whether the request is to be approved or declined, in which case the card application will either proceed with the transaction or reject it. It is important to note here that it is optional for a card issuer to examine a customer's platform state as part of their decision making process. Indeed the Tag Length Value (TLV) encoding mechanism used in EMV makes it very easy for a card issuer to ignore any extraneous information from a transaction referral request. Significantly, if the card issuer does decide to take a customer's platform state into consideration, the bank itself would not need any Trusted Computing facility to examine these characteristics. It would only need to be capable of hashing some supplied records and verifying a signature (as we shall see in Section 7). Indeed this functionality could be provided by a third party facility or could very well be preformed by the acquiring host.

7 Installing and Instantiating an e-EMV card

This section explains in detail the procedures and processes involved in establishing an e-EMV card within a TPM-enabled host.

7.1 Account Activation

The account activation process for of an e-EMV card comes as a result of TPM-enabled platform becoming a member of an card issuer controlled group. This process is mirrored for the merchant server account activation procedure with respect to a merchant's acquirer. In both cases, account activation is achieved by successfully obtaining/generating a credential issued for an AIK public key. This credential could be in the form of X.509 certificate issued by a Privacy CA or it could be as a result of a platform performing a DAA Sign operation on a public AIK component (see Section 5.1.4). In either case the end result is similar. The customer will have their account activated, later allowing them to demonstrate physical/logical possession of an active card within the system. Through the establishment of customer-centric credentials embedded within a platform, the customer will be able to attest to the existence of a key capable of replicating EMV's SDA/DDA/CDA key functionality. Here simulating SDA/DDA/CDA is dependant on the card issuer offering Privacy CA or DAA services to its customers.

As part of account activation there is an additional requirement for a customer to obtain an ICC Public Key Certificate. Within the EMV specifications, the ICC Public Key Certificate holds a particular meaning as it contains a slew of vital information, most notable of which is the Application PAN and cardholder name. What follows is two different approaches for a customer to obtain this ICC PK Certificate.

7.1.1 Privacy CA Approach

The Privacy CA approach is used to obtain a credential from a card issuer which will later be used to validate signed platform metrics demonstrating both possession, and correct usage of, a valid e-EMV card.

This process involves the generation of an AIK IC key pair (S_{IC} and P_{IC} for the private/public portions respectively) and having the public portion incorporated into an ICC PK Certificate. The process involved in the generation of a new ICC key within a platform maps to the generation of an AIK key within a TPM. That is, as a result of performing the TPM_MakeIdentity command [26, pp.147] which results in the creation a new AIK under the control of the SRK. The P_{IC} portion, various platform credentials as well and customer authentication data is encrypted with the issuing hosts public key (see Section 2) and sent to the Privacy CA. If after authenticating a particular customer and satisfying itself that the request is coming from a genuine TPM, the Privacy CA will issue an ICC PK Certificate (AIK credential) to the customer's platform.

This credential can then be used to provide evidence of card activation within the system. Additionally the ICC PK Certificate could be further enhanced through X.509 v3 extensions. In this regard, it is possible to add key and policy information to the credential, such as setting a private key usage period under which the AIK signing key will operate.

7.1.2 DAA Approach

The DAA enrollment procedure occurs exactly as laid out in Section 5.1.4. However, here the DAA issuer is instantiated by an card issuer controlled issuing host. Pursuant to a successful completion of the DAA join procedure, the customer will be able to demonstrate that their account has been activated with respect to a particular card issuer.

It is important to note here that the usage of ζ in the DAA signing process confers certain anonymity properties to a customer. If the value of ζ is changed for every transaction a customer instigates then they can prove they were issued a valid credential yet remain anonymous in the signing and verification procedure. In itself this property doesn't hold much value in traditional CNP style transactions as typically the customer provides copious identifying information. Its real advantage comes in enabling future (possibly anonymous) P2P commerce and in the provision of digital goods, as well as allowing a delineation between browsing and ordering phases.

Subsequent to a successful completion of either the DAA join procedure or upon receipt of a credential from a Privacy CA, the card issuer activates the customer's account within their systems. This then enables the soon to be downloaded e-EMV application to be used in ensuing financial transactions.

There is one issue, however, that pervades both the Privacy CA and DAA approaches. That is, the use of an EK in platform authentication. It remains unclear how much meaning an EK would have to an card issuer or acquirer as neither would typically have any interaction with platform manufacturers. However, there is no reason why a separate non-migratable TPM key pair couldn't be used. This key can be sent with customer authentication details to the issuing host in the same way as an AIK public key is sent in the Privacy CA approach. The important part here, isn't so much that a request is coming from a particular TPM, it's that the request is coming from a particular customer. We observe customer authentication in-and-of itself isn't necessarily a trivial task. We assume some sort of proof of knowledge examination, for example, knowledge of a shared secret created in the pre-enrollment stage. However, the actual mechanism used is orthogonal to the discussion.

7.2 Secure Application Delivery

The delivery of the combined set-up and interface utility to a customer’s platform is an interactive process between a customer and their card issuer. The delivery of the program itself necessitates a number of checks to ensure the binding between a customer and their platform. This is because the set-up utility requires the inclusion of a unique ICC Master Key per card issuance and hence cannot be generated by the platform itself (see Section 4.2.2). Instead the ICC Master Key needs to be injected into the platform as part of the application delivery process. In addition to the secure delivery of the application, there is an underlying customer-driven requirement to ensure the authenticity and the “behaviour” of the application once delivered.

The dual concerns of authenticity and “behavior” can typically be addressed using what is termed a validation credential in the Trusted Computing literature (see Section 5.1.6. In this way runtime metrics can be compared against known good values to assure customers that their application is performing as intended.

The issue of secure delivery of an application using Trusted Computing has previously been examined in the context of conditional access in mobile systems [15, 16] as well as being alluded to in a number of the TCG’s publications, such as [19]. In our discussion, the delivery of an application to a customer’s platform, as well as the corresponding requirement of securely storing the application upon receipt, can be handled as follows:

These requirements are fulfilled by creating a Certified Migratable Key (CMK) pair then “sealing” the application to the the customer’s TPM. Sealed messages in the context of Trusted Computing are messages that are bound to a set of platform metrics specified by the message sender and thus can only be opened when the platform is in a certain state. This is normally achieved with the `TPM_Seal` [26, pp.54] or the `TPM_Sealx` command [26, pp.75]. However, the notion of sealing an application to a platform typically has local significance only. We require the same functionality as provided by the `TPM_Seal` command yet slightly different semantics. The `TPM_Seal` command works by the verifier (card issuer) sending a `TPM_PCR_INFO` [25, pp.68] structure to the remote platform (customer). However, as our requirements dictate that the keys be migratable (in order to allow a customer to port their card to another platform) the option of `TPM_Seal` is no longer viable. This is because the TPM specifications explicitly forbids the use of non-migratable keys in sealing. Our requirements also differ in that the card issuers be allowed to “seal” the data on their platform and send it to the requesting customer’s remote platform.

The set-up phase of this can be illustrated programmatically as in Algo-

rithm 1.

Algorithm 1 Set-up Phase

- 1: $IC_Key := TPM_CMK_CreateKey (TPM_Auth_Always, digestAtCreation, digestAtRelease, IssuingBank_{pubkey})$
 - 2: $TPM_Certify_Info2 := TPM_CertifyKey2(IC_Key)$
 - 3: $Signature := AIK/DAA_{Sign}(Hash(TPM_Key_{pubkey}))$
-

Here the customer creates a new certified migratable key specifying both current and future platform states required for key retrieval, as well as specifying that the key will always require authorisation. Additionally the Migration Authority is set via the card issuer’s public key (See Appendix A for a discussion of CMK migration).

The next step involves having the newly generated key certified by using the `TPM_CertifyKey2` command (which is permissible as the key is a CMK) and finally having a hash of the public part of this key signed using DAA⁴ or a Privacy CA certified key⁵ as outlined in Section 7.1. In this sense the customer is effectively producing a signature over the public CMK using a signature key that the card issuer knows to be bound to a particular customer. In this setting `IC_Key` can provide the same level of assurance as results from the `TPM_Seal` command. This is because during the generation of the key both the “`digestAtCreation`” and “`digestAtRelease`” parameters are fully specified. In this case the `digestAtRelease` is semantically equivalent to sealing.

A more fully specified exchange protocol is as follows (here R_x is a random number, IB is the card issuer and C is the customer):

- 1: $IB \rightarrow C : R_{ib} \parallel IB_{pubkey} \parallel [Platform\ Attestation]$
- 2: $C \rightarrow IB : IB_{pubkey}(TPM_Certify_Info2 \parallel IC_Key_{pubkey} \parallel Signature \parallel R_{ib} \parallel R_c)$
- 3: $IB \rightarrow C : R_c \parallel IC_Key_{pubkey}(application \parallel Cert_{TPM_Key})$

1. In step 1 the card issuer sends a challenge and optionally its own platform state to the customer in conjunction with its public key in its certificate.

⁴DAA is capable of signing 160-bits of externally generated data or AIK keys depending on the status of a 1-bit flag

⁵AIK keys themselves are incapable of signing the hash result directly, instead the signature would be as a result of incorporating the hash as measured data in a PCR register.

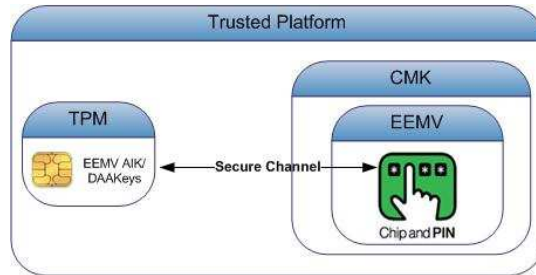


Figure 4: e-EMV Trusted Platform resident application

2. In step 2 the customer verifies this platform state (if present) as well as the public key certificate. If all goes well, the customer returns the public portion of the key it generated during the set-up phase along with the requisite information specifying the control policy for the private portion of the key. In addition to this it sends its own signed platform metrics.
3. In step 3, the verifier examines the received data. If the storage semantics for the private key match its own security policy then it sends the application (with embedded ICC Master Key) to the customer.

The final phase of the protocol for secure storage is application retrieval. The basic algorithm is as follows:

Algorithm 2 Message Retrieval

```

if Platform state == digestAtRelease && incomingAuth == objAuthData
then
    Retrieve message
else
    Fail
end if

```

Providing that the current platform state matches the requirements for the “sealed” data and the incoming authorisation data matches the authorisation data set for the private key during the set-up phase, then unseal the data. If neither of these two conditions are met then the message should remain sealed until both conditions can be fulfilled.

Ensuant from a successful delivery of the e-EMV set-up bundle (Figure 4), the customer is now capable of utilising their SDA/DDA/CDA compliant card on their TPM-enabled platform.

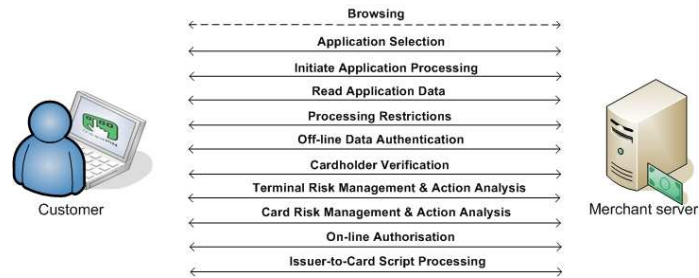


Figure 5: e-EMV Customer - Merchant Interface

8 e-EMV in Operation

Transaction processing in e-EMV follows the EMV message flows from section 4.1 and are illustrated in Figure 5. The following provides a detailed mapping of the EMV transaction architecture as applied to e-EMV.

8.1 Customer - Merchant Interaction

Customer to Merchant interaction in the virtual environment mirrors transaction processing at POS terminals in which the transaction flows of an EMV transaction can be mapped to an e-EMV transaction as follows:

1. **Application Selection:** Typically, application selection occurs in response to a terminal initiated command to select the appropriate EMV application from a multi-application card. In this case we assume a single application instance in which customer/merchant application matching is dependent on finding a suitable AID. The AID in this case can be either a set of validation credentials supported by the platform or a reported set of one or more PCR values in the form of a platform attestation, representing a valid application execution.
2. **Initiate Application Processing:** Following on from the above, the customer's application commences application processing. In the physical world this would yield a response in which the card returns its application interchange profile and its application file locator. However, in this setting as the cards are virtualised and free from the constraints of typical ICC cards, all e-EMV cards would represent DDA/CDA capabilities. Additionally, as part of this step the merchant terminal plug-in provides to the e-EMV application any terminal-related information pertaining to the business environment at the point of service.

An example of such information would be terminal capabilities or the country in which the terminal is operating.

3. **Read Application Data:** The steps involved in issuing one or more READ RECORD commands in this setting is relatively redundant as we assume the application is executing solely in main memory on a customer's platform.
4. **Processing Restrictions:** This mandatory step is performed by the merchant and doesn't require any direct interaction with the customer's execution environment. This step is primarily concerned with judging the compatibility of the e-EMV application with that of the terminal application. This process can be handled in an e-EMV environment through a process of reconciliation between terminal supported applications and customer supplied validation credentials/platform attestation.
5. **Off-line Data Authentication:** This step, corresponding to a POS terminal issuance of an INTERNAL AUTHENTICATE command is achieved through the TPM's Attestation mechanism. The three options for off-line data authentication available in EMV-compliant cards are: SDA, DDA and CDA.
 - (a) SDA - SDA is relatively simple process whereby a validation credential could be generated (by an card issuer) providing metrics for a correctly functioning e-EMV card application. These metrics can then be compared against the runtime PCR metrics reported in an attestation challenge to a merchant. An outline of this process is as follows:
 - i. An merchant server requests one or more PCR values corresponding to the PCRs to which the e-EMV application is bound. In addition to this request the merchant attests its own platform metrics for digestion by the customer's platform.
 - ii. The customer's platform examines the attested metrics and determines the suitability of the merchant for adherence to certain desirable policies, such as the PCI standards for card processing. If satisfied, a customer's platform agent culls the SML for the events responsible for generating the requested PCR values.

- iii. The TPM loads the AIK/DAA signing key using a customer provided password. This loaded key then signs the requested registers using an AIK. This will either be the same AIK for which a certificate was obtained during the e-EMV account activation stage (as per the Privacy CA model) or a new AIK over which a DAA signature will be generated.
 - iv. The platform agent then obtains various credentials (the platform credential, the conformance credential, attestation identity credential, validation credentials) that vouch for the TPM. These credentials, along with the relevant portion of the SML and the requested signed PCR values are returned to the merchant server for verification.
 - v. The merchant then examines the credentials, checks signatures⁶ and compares a hash of the SML entries to the attested PCR values. If they match the verifier can be sure as to the current state of the customer's platform. In effect achieving SDA.
- (b) DDA - DDA by contrast requires a slight modification to the SDA mechanism as outlined in step 5(a)iii of the above. During the operation of TPM_Quote in which the PCR registers are signed, the customer incorporates a merchant supplied 160-bit challenge to the attestation. This challenge takes the place of the operand *externalData* which is of type TPM_NONCE in the PCR attestation process.
- (c) CDA - CDA consists of an e-EMV generated dynamic signature (similar to DDA but includes an AC generation) followed by verification of this signature by the merchant. In the EMV POS environment this would consist of a ICC private key signature over the Signed Dynamic Application Data of which a TC or ARQC and an unpredictable number (as per DDA) are integral parts. Incorporating this into a Trusted Platform environment isn't necessarily any more complicated than either the SDA or DDA approaches. All that is required is having the e-EMV application incorporate AC generation as measured data within the system. That way, the generation of an AC, along with its data output (TC/ARQC) can be integrated into a PCR register. The actual examinable out-

⁶There may be a situation in which a merchant is unfamiliar with certificate authority espoused in ICC public key certificate. However, provided each card issuer is itself a link in a chain traversing back to globally recognised root CA, for example Visa or Mastercard, then this verification can be as a result of certificate chain traversal

put, that is the TC or ARCQ, is then correspondingly appended to the SML and a signature is generated over the representative PCR values. In this way the merchant, upon receipt of the attestation bundle can examine the SML for the inclusion of an AC and verify its veracity through a simple SHA-1 hash and signature comparison.

6. **Card holder Verification:** In the typical POS EMV operation, this step allows the terminal to verify the authenticity of a cardholder. This authentication is based on the card and terminal both supporting a particular CVM. In addition to a CVM verifying a customer, some CMVs, particularly PIN authentication, are used as a means of authorising a transaction.

Through the use of Trusted Computing functionality we can make PIN authentication and authorisation implicit within a platform. This is achieved through the TPM key authorisation mechanism whereby certain keys require 20 bytes of authorisation data to be supplied prior to being loaded. In the architecture presented so far, we have instances of two different key types, Attestation Identity Keys and Certifiable Migratable Keys. Both of these require authorisation, in the form of TPM_AUTHDATA, to be securely communicated to a TPM. In our architecture, this authentication data will be communicated over a secure Object-Independent Authorization Protocol (OIAP) session [24, pp.62]. The use of an OIAP session allows authorisation information to be sent TPM without revealing the data on the channel over which it is sent. This in effect is an adjunct to the assumption of a Trusted Path provided by the OS in Section 5.5.

The ability to actually use a key as part of a transaction demonstrates to a merchant that a CVM method has occurred successfully. In the case of the CMK, to actually load the e-EMV application, and in the case of the AIK/DAA key to sign platform metrics. By verifying the state of the platform, a merchant can be assured that only a valid customer would be capable of using the AIK/DAA key generated in the account activation stage.

7. **Terminal Risk Management and Action Analysis:** The purpose of of terminal risk management and terminal action analysis is to protect the issuer, acquirer and payment system from fraud. The variety of measures used in terminal risk management, such as floor limits, random transaction selection and velocity checking would remain largely

unaffected in the transition to an e-EMV payment architecture. All such services would be replicated in the merchant terminal plug-in.

8. **Card Risk Management and Action Analysis:** Details of card risk management are proprietary to card issuers and are outside the scope of the EMV specification and as such would remain proprietary to individual card issuers within this system.
9. **On-line Authorisation:** During normal transaction processing the merchant terminal may decide to proceed with an on-line check, this again remains unaffected in an e-EMV environment.
10. **Issuer Script Processing:** Issuer Script processing is perhaps the most difficult EMV feature to replicate in an e-EMV setting. It would be impossible for an card issuer to force an update on a customer's platform without gaining their consent. This remains an open issue in an e-EMV environment.

8.2 Application Cryptograms

As we saw from Section 4.2.2 application cryptograms are generated by an ICC using the session keys derived from the ICC Master Key. These session keys are then used to protect transaction messages. In our architecture, in conjunction with the information typically sent in the AC message, the e-EMV application appends the current platform state as witnessed by their issuer validated signing key. After examining the AC message as well as the supplied state the verifier, be they merchant or card issuer, can make a decision as to whether or not to proceed with a transaction. In the instance where an AC is an ARQC, as mentioned in Section 6.2, the TLV encoding scheme used in EMV allows the issuer to ignore extraneous information if they so choose. The card issuer can then respond with an ARPC indicating whether the transaction should be approved or declined, in which case a TC or an AAC will be generated by the customer's platform.

As we have seen in Section 4.2.2, an important aspect in the generation of ACs is the use of the ATC as diversification data. Trusted Computing can be used to model the ATC using monotonic counters [24, pp.80]. Monotonic counters provide an ever-increasing incremental value which can provide the exact same functionality as described for EMV's ATC. For every transaction that is initiated by a customer, their TPM can increment a monotonic counter associated with their e-EMV card. Indeed we can model any additional EMV counters using this Trusted Computing facility.

9 An Application of Trusted Computing to CNP transactions

9.1 Pseudonymous Credit Cards

Pseudonymous credit cards are an interesting corollary to the DAA account activation process as highlighted in section 7.1, and can be seen as supplementing the ideas presented in [7]. In this respect we aim to replace PAN account information with a representative pseudonym, both in the context of authenticating as a valid member of an card issuer controlled group and in producing a payment token in the form of an EMV TC.

As mentioned previously, by itself this pseudonymous property doesn't hold much value in traditional CNP transactions, as often the customer provides ample identifying information. However, we can gain some traction in helping enabling pseudonymous P2P commerce, particularly in the provision of digital goods as well as allowing a possible delineation between browsing and ordering phases of a transaction.

During the registration procedure the platform must reveal a pseudonym N_I of the form ζ_I^f . Here the quantity ζ_I^f is set by the customer's card issuer and allows them to associate a "handle" with a customer as well as to check for rogue platforms during payment resolution. This pseudonym, N_I , can be used in the derivation of the ICC Master Key — instead of using a PAN as specified in Annex A1.4 of [12]. In this way a customer can convince a merchant that they are indeed in possession of a valid card through a zero knowledge proof, but without revealing a particular account number which could be linked to subsequent transactions. In this way we can provide anonymity through unlinkability. This can further be delineated between unlinkability between the browsing and ordering phases, where a customer may insist on choosing random base names (ζ) during their browsing and ordering phases, or even use a fixed base name in the form of a derivation of a merchant's name raised to their secret f . In this way a customer is linkable to a single merchant but not across various merchant stores. This is advantageous in the situations where customers can gain loyalty bonuses for continued patronage of a merchant site.

When it comes to generating a transaction certificate to complete a particular payment, the customer can do so as normal, as the ICC Master key used in the generation of an AC is a diversification of the customer's handle with their card issuer. This combined with a DAA signature covering the TC and current platform metrics can uniquely identify a particular customer, but only to the bank to which they belong. Additionally, as the recommended

minimum set of data elements for AC generation, as laid out in Table 25 of [12] details no mention of uniquely identifying information, the merchant details can remain separate to a payment request. Thus achieving a result similar to that of the delineation of OI and PI in a SET environment.

10 Conclusions

The proliferation of the Internet as an avenue for electronic commerce, in the form of CNP transactions, has seen something of a popular explosion in recent years. However, with this explosion has come a number of issues, most notable of which is the high level of transaction chargebacks (several times the system average) seen in CNP transaction. The level of fraud, whilst not yet reaching endemic proportions, has raised significant concern amongst the card processing community. This has led to a number of proposals being put forward to try and address this problem.

As we saw from Section 3, these proposals fail to address certain key issues in securing on-line CNP transactions. There is no way of ascertaining if the customer presenting a particular card (typically in the form of a PAN) is in possession of the card at time of purchase or indeed if they are the valid owner of the card. Of the proposals presented, only SSL has enjoyed any real quantifiable success, and from our analysis, is actually the weakest in terms of authenticating customers prior to transaction processing.

This report proposed a new security architecture for securing CNP transactions. By creating software based EMV cards running on Trusted Platforms we replicate many of the features of standard EMV-compliant cards for use in CNP transactions. Through the account activation procedure outlined in Section 7.1 we established how we can remotely demonstrate possession of a card within our system. We showed how card ownership can be demonstrated through the use of an OIAP session. We also showed how EMV transaction messages can be mapped to e-EMV transaction messages. In this regard, we demonstrated how EMV keys can be generated and bound to a particular TPM-enabled platform. We also showed how an e-EMV card can be ported from one TPM-enabled platform to another.

In this respect we can achieve a significant improvement in the level of security afforded in both initiating, completing and processing CNP transactions. Indeed, the e-EMV infrastructure presented could be used as a basis for establishing both fungible and non-fungible payments. It is not necessarily restricted to card payments although this was the basis for this research.

11 Appendices

A CMK Generation and e-EMV Migration

Migration in the context of Trusted Computing is the process through which a migratable key moves from one TCG-compliant platform to another. This allows the new platform to function exactly as the old one with respect to key usage. The `TPM_CMK_CreateKey` command [26, pp.93] is responsible for the generation a new certified migratable key within a platform. This command takes as input the public key of an Migration Selection Authority (MSA), in this case the card issuer.

When it comes to the actual migration of a CMK a number of entities may be involved in the migration, foremost of which are the MSA and possibly a Migration Authority (MA), both of which are assumed to be under the control of the card issuer in this particular scenario. In this setting the MSA controls the migration whilst the MA handles the migrated key. The actual migration of a CMK is as a result of calling `TPM_CMK_CreateBlob` [26, pp.99] and requires the authorisation of the MSA as well as that of the `TPM_Owner` (customer).

This process begins with the creation of an owner authorised `TPM_MIGRATIONKEYAUTH` structure [25, pp.36] using `TPM_AuthorizeMigrationKey` command [26, pp.85]. Inherent in this structure are the destination migrationKey, the migrationScheme (which must be set to `TPM_MS_RESTRICT_APPROVE` or `TPM_MS_RESTRICT_APPROVED`) and a digest of `tpmProof`. The approval of an MA to agree to facilitate the migration of the key comes in the form of a signature over a `TPM_CMK_AUTH` structure [25, pp.40]. This structure contains the digest of the MA public key, the digest of the destination public key and a digest of the public portion of the key to be migrated. Finally, the TPM Owner authorises the MA's approval using `TPM_CMK_CreateTicket` [26, pp.97] and generates a signature ticket. This signature ticket is then passed, along with the TPM Owner's authorisation, the MA's authorisation and the MSA's approval to the `TPM_CMK_CreateBlob` command [26, pp.99] allowing the key to be migrated to another platform.

In this way the the wrapped e-EMV application and the CMK which wraps it can be ported from one TPM-enabled platform to another TPM-enabled platform of the customers choosing. This occurs with one important caveat, the migration itself is a strictly controlled process requiring both customer and card issuer authorisation. In this way a card can not be extracted from a platform without explicate customer consent. After migration of the CMK and the application, the customer would re-run the account activation

stage in order to establish the card within the new platform. This is a necessary step as neither an AIK nor a DAA value f are migratable structures. The card issuer would then update its system to reflect a new card being linked to an existing customer PAN, much in the same way as secondary cards are linked to primary account numbers.

References

- [1] M. Al-Meather and C. J. Mitchell. Extending EMV to support murabaha transactions. In *Proceedings of the Seventh Nordic Workshop on Secure IT Systems - Encouraging Cooperation*, pages 95–108. NORDSEC 2003, 2003.
- [2] Tiago Alves and Don Felton. Trustzone: Integrated hardware and software security. <http://www.arm.com/products/CPUs/arch-trustzone.html>, July 2004.
- [3] APACS. Card fraud the facts 2005, April 2005.
- [4] APACS. Card fraud the facts 2006. http://www.apacs.org.uk/resources_publications/documents/FraudtheFacts2006.pdf, April 2006.
- [5] Visa International Service Association. 3-D Secure™ Protocol Specification: Core Functions. <http://international.visa.com/fb/paytech/secure/main.jsp>, July 2002.
- [6] Visa International Service Association. 3-D Secure™ Protocol Specification: System Overview. <http://international.visa.com/fb/paytech/secure/main.jsp>, May 2003.
- [7] S. Balfe, A.D. Lakhani, and K.G. Paterson. Securing peer-to-peer networks using trusted computing. In C.J. Mitchell, editor, *Trusted Computing*, pages 271–298. IEE Press, 2005.
- [8] S. balfe and K.G. Paterson. Augmenting Internet-based Card Not Present Transactions with Trusted Computing: An Analysis. Technical report, Technical report RHUL-MA-2006-9, (Department of Mathematics, Royal Holloway, University of London, 2005). <http://www.rhul.ac.uk/mathematics/techreports>.
- [9] J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *Security in Communication Networks, Third International Conference, SCN 2002*, volume 2576 of LNCS, pages 268–289. Springer Verlag, 2003.
- [10] EMVCo. *Book 3 - Application Specification*, 4.0 edition, December 2000.
- [11] EMVCo. *Book 1 - Application independent ICC to Terminal Interface requirements*, 4.1 edition, May 2004.

- [12] EMVCo. *Book 2 - Security and Key Management*, 4.1 edition, May 2004.
- [13] EMVCo. *Book 3 - Application Specification*, 4.1 edition, May 2004.
- [14] EMVCo. *Book 4 - Cardholder, Attendant, and Acquirer Interface Requirements*, 4.1 edition, June 2004.
- [15] E. Gallery and A. Tomlinson. Conditional access in mobile systems: Securing the application. In *First International Conference on Distributed Frameworks for Multimedia Applications (DFMA'05)*, pages 190–197. IEEE, 2005.
- [16] E. Gallery, A. Tomlinson, and R. Delicata. Application of trusted computing to secure video broadcasts to mobile receivers. Technical report, Technical report RHUL-MA-2005-11, (Department of Mathematics, Royal Holloway, University of London, 2005). <http://www.rhul.ac.uk/mathematics/techreports>.
- [17] Trusted Computing Group. *TCG PC Specific Implementation Specification*, 2003.
- [18] Trusted Computing Group. Trusted computing: Opportunities and challenges. <https://www.trustedcomputinggroup.org/downloads/tcgpresentations/>, 2004.
- [19] Trusted Computing Group. *Mobile Phone Work Group Use Cases*, 2.7 edition, 2005.
- [20] Trusted Computing Group. *TCG Infrastructure Working Group Reference Architecture for Interoperability (Part I)*, 1.0 revision 1 edition, 2005.
- [21] Trusted Computing Group. *TCG Trusted Network Connect TNC Architecture for Interoperability*, 1.0 revision 4 edition, 2005.
- [22] Trusted Computing Group. *TCG Trusted Network Connect TNC IF-IMC*, 1.0 revision 3 edition, 2005.
- [23] Trusted Computing Group. *TCG Trusted Network Connect TNC IF-IMV*, 1.0 revision 3 edition, 2005.
- [24] Trusted Computing Group. *TPM Main: Part 1 Design Principles*, 1.2 revision 85 edition, 2005.

- [25] Trusted Computing Group. *TPM Main: Part 2 Structures of the TPM*, 1.2 revision 85 edition, 2005.
- [26] Trusted Computing Group. *TPM Main: Part 3 Commands*, 1.2 revision 85 edition, 2005.
- [27] Trusted Computing Group. *TCG Generic Server Specification*, 2005 Revision 0.8.
- [28] Trusted Computing Group. *TCG Software Stack Specification Version 1.2 Level 1*, 2006.
- [29] E.V. Herreweghen and U. Wille. Risks and potentials of using EMV for internet payments. In *In Proceedings of the First USENIX Workshop on Smartcard Technology*. USENIX, May 1999.
- [30] Intel. Lagrande technology architectural overview. <http://www.intel.com/technology/security/>, September 2003.
- [31] MasterCard International. SecureCodeTM Merchant Implementation Guide. <http://www.mastercardmerchant.com/securecode/>, March 2004.
- [32] V. Khu-Smith and C.J. Mitchell. Using EMV cards to protect e-commerce transactions. In *Lecture Notes in Computer Science*, volume 2455 of *E-Commerce and Web Technologies: Third International Conference*, page 338, January 2002.
- [33] J. Leyden. Chip and pin hits 8 million cards.
- [34] M. and T. Wobber. A logical account of NGSCB. In *24th IFIP WG 6.1 International Conference on Formal Techniques for Networked and Distributed Systems*, volume 3235 of *LNCS*, pages 1–12. Springer Verlag, 2004.
- [35] P. Meadowcroft. Combating card fraud. <http://www.scmagazine.com/uk/news/article/459478/combating+card+fraud/>, January 2005.
- [36] D. O'Mahony, M. Peirce, and H. Tewari. *Electronic Payment Systems for E-Commerce 2nd edition*. Artech House, 2001.
- [37] M. Peinado, Y. Chen, P. England, and J. Manferdelli. NGSCB: A trusted open system. In *Proceedings of the 9th Australasian Conference on Information Security and Privacy*, volume 2738 of *LNCS*, pages 86–97. Springer Verlag, 2004.

- [38] C. Radu. *Implementing Electronic Card Payment Systems*. Artech House, 2002.
- [39] IBM Global Services. IBM Global Business Security Index Report, February 2005.
- [40] Visa and Mastercard. Payment card industry data security standard. http://usa.visa.com/download/business/accepting_visa/ops_risk_management/cisp_PCI_Data_Security_Standard.pdf, 2004.
- [41] J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, and D. Spence. RFC2904 – AAA Authorization Framework, 2000.
- [42] J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, and D. Spence. RFC2905 – AAA Authorization Application Examples, 2000.
- [43] J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, and D. Spence. RFC2906 – AAA Authorization Requirements, 2000.
- [44] K. Zetter. Cardsystems' data left unsecured. <http://www.wired.com/news/technology/0,1282,67980,00.html>, 2004.