Augmenting Internet-based Card Not Present Transactions with Trusted Computing

Shane Balfe and Kenneth G. Paterson

Technical Report
RHUL-MA-2006-09-v2
07 March 2008

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

Abstract

In this paper, we demonstrate how Trusted Computing technology can be used to enhance the security of Internet-based Card Not Present (CNP) transactions. We take a pragmatic approach, focusing here on exploiting features of Trusted Computing as it is being deployed today. Thus we rely only on the presence of client-side Trusted Platform Modules, rather than upon the "idealised" deployment in which Trusted Computing functionality is fully integrated with OS and CPU, and which still seems to be a distant prospect. In essence, our approach uses features of the Public Key Infrastructure that is inherent in Trusted Computing to build lightweight client-side enrollment and certification processes; public key certificates are then used to underpin authentication for CNP payments. Using this approach we demonstrate how Trusted Platform Module (TPM) enabled platforms can integrate with SSL and 3-D Secure. We discuss the threats to CNP transactions that remain even with our enhancements in place, focussing in particular on the threat of malware, and how it can be ameliorated.

1 Introduction

The Internet as an avenue for card-based commerce has seen something of a popularity explosion in recent years. In the UK alone, on-line shopping has become a multi-billion pound industry and in 2004 accounted for nearly 11 pence out of every £1 spent using credit cards. However, this particular form of commerce, typically referred to as Card Not Present[1] (CNP) transactions, whilst commonplace, is currently far from secure.

A recent report by the Association for Payment Clearing Services (APACS) on card fraud [3] showed that Internet-based CNP transactions accounted for 36% of all card fraud perpetrated in 2006 in the UK (up from 27% the previous year). This translated into £154.5 million in losses for card issuers and merchants. The 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 these transactions. However, the vast majority of Internet-based payments are secured using a single protocol suite, namely SSL, to protect card account information.

Unfortunately, this usage of SSL is not a panacea for enabling secure Internet-based CNP transactions. SSL was not designed as a payment protocol but instead adopted as a *de facto* standard for securing CNP transactions.

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

Indeed, the use of SSL in CNP transactions has a number of shortcomings. 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. For example, SSL is used only in relation to securing the payment channel; there is no guarantee that the customer owns the account number being proffered in a particular payment transaction. 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.

To address some of these inadequacies other proposals for securing CNP transactions, such as the *i*KP protocols [8] and their successor, the Secure Electronic Transaction (SET) protocol [27] have been proposed. However, whilst offering additional security benefits over an SSL-based approach, neither protocol suite has seen wide-spread adoption. One relatively new proposal, however, namely 3-D Secure [4], appears to becoming widely deployed. 3-D Secure is an optional adjunct to the SSL-based approach and attempts to provide cardholder authorisation for CNP transactions by requiring customers to authenticate themselves prior to transaction processing. This authentication forms an ancillary step to regular merchant checkout processing where, after receiving a customer's PAN and CSC, a merchant site redirects its customer to a 3-D Secure Access Control Server (ACS) to which the customer authenticates. If successfully authenticated, the ACS informs the merchant who then proceeds with regular transaction processing based upon the previously supplied account details. This approach aims to tackle the fraudulent acquisition of card account details for use in CNP transactions by providing a delineation between card authentication data and customer authentication data. However, this approach has only limited security benefits in the face of the threat of malware such as trojans and keystroke loggers, a threat which is increasing at a frightening rate. According to [29], between the period July and December 2006, five of the top ten new malicious code families detected were trojans with keystroke logging threats accounting for 79 % of confidential information leakage threats by volume of reports. However, perhaps most worrying is statistic that home users now account for 93 % of *all* targeted attacks [29]. In this setting a piece of malicious software residing on a customer's platform could capture user authentication credentials and manipulate transactions (including possibly instigating new transactions).

To address this issue there has been a recent development to strengthen 3-

D Secure's authentication process through integrating with EMV[2] chip cards. This approach involves the use of "unconnected" card readers which, when interacting with a customer's physical card, generate a one-time passcode on a per-tranaction basis [21]. This passcode would then be used instead of a customer-supplied password for 3-D secure authentication. However, this approach suffers from the costs associated with distributing card readers to end-users, and as yet there are no publicly available specifications detailing the precise operation of such a system. Additionally, there have been recent reports of time-of-check to time-of-use attacks on similar two-factor authentication schemes [19].

1.1 Our Contributions

This paper examines how Trusted Computing can be used to enhance the security of existing protocols (SSL and 3-D Secure) in the provision of secure CNP transactions. In doing so, we highlight a number of well known weaknesses in their (unmodified) deployment and show how they can be addressed using Trusted Computing.

More precisely, we operate from the sole assumption that client platforms are equipped with Trusted Platform Modules (TPMs) having limited but trusted cryptographic functionality. The real-world applicability of this approach is demonstrated by the market-penetration of TPM-enabled platforms: currently available sales figures for 2005 [11] showed estimates of 32% of all notebook systems shipped that year being TPM enabled. This figure is expected to nearly triple by the end of 2007 with similar growth expected in other device types. We use the TPM's trusted capabilities to build lightweight client-side enrollment and certification processes. These effectively bind a platform, and by extension its owner, to a particular card. The resulting public key certificates and TPM signing capabilities are then used to underpin authentication for CNP payments. The card involved may be a plastic one already in the hands of the customer, or a virtual card created especially for use in on-line transactions.

One of the most salient issues in the development of our approach is the problem of customer enrollment, during which a customer/card binding is established. We examine different system architectures and discuss the pros and cons of their associated enrollment procedures. Another major issue is the increasing threat of malware (or, more specifically, crimeware) and its ability to create spurious transactions or modify on-going ones. We also study the malware/crimeware threat, explaining how it can be reasonably

[2] http://www.emvco.com/

addressed within our architecture using the secure attention sequences that are a mandatory part of the TPM.

An oft-cited reason for the failure of SET was its reliance on PKI and client-side certificates. Our approach would appear to suffer from the same problems. Given the rise of Internet-based CNP transactions and the associated fraud levels, we believe that the economic conditions are now far more ripe for the deployment of solutions based on client-side certification. Moreover, we are able to take advantage of the TPM certification infrastructure to support Trusted Computing. This PKI needs only to be augmented with certain CA functionality, provided by a card issuer in our approach.

We stress here that we do not need to make any further assumptions about the deployment of Trusted Computing in order for our approach to provide useful security benefits for CNP transactions. In particular, we need not rely on a full-blown deployment of Trusted Computing in which the TPM is fully integrated into the host platform's processor, boot process and trusted Operating System.

1.2 Related Work

The idea of using Trusted Computing to enable client-side certification has previously been discussed in [13, 1, 7] as well as in the as-yet-unpublished Trusted Computing Group's TLS extensions for carrying attestations. However, none of this work takes into consideration the threat posed from malware nor the infrastructural requirements necessary to support client-side certification. The threat from malware is examined in greater detail in [10, 17, 18]. There, Virtual Machines (VMs) are used to constrain the use of malware to an individual VM "compartment". These authors also suggest using visual cues as to the trustworthiness of the VM with which an end-user interacts, an idea originating in [6]. Other related work includes the use of Trusted Computing as an adjunct to securing connected card readers for generating digital signatures, presented in [28] and [6]. However, both approaches, much like the unconnected card reader proposal outlined above, suffer from costs associated with the provision of card readers to end users. Additionally, both proposals assume the presence of trusted software to interact with the readers.

Recent works that take a pragmatic approach to trusted computing include [26] and [20]. In [20], McCune *et al.* attempt to address the problem of user-level malware in the absence of Trusted Computing processor and chipset support. They propose the use of an external trusted mobile device to establish an encrypted and authenticated channel between the user and a TPM host. In [26], Sarmenta *et al.* implement virtual monotonic coun-

ters to prevent replay attacks on an untrusted machine aided solely by TPM support. They suggest a number of possible applications of their approach, such as n-time use keys and n-copy migratable objects.

1.3 Paper Outline

This paper is structured as follows. In Section 2, we introduce the steps involved in a CNP payment clearing process as well as a two of the most widely deployed protocols to protect CNP transactions. In Section 3, we introduce core concepts of Trusted Computing that we will later apply to securing CNP transactions. In Section 4, we examine the issue of customer enrollment with particular emphasis on the establishment of customer-centric credentials within a TPM-enabled platform. Section 5 examines the role these TPM-enabled customer credentials can play in supplying additional security to SSL and 3-D Secure. Finally, we conclude with Section 6.

2 CNP transactions and the Internet

This section begins with an overview of the generic four corner model used in card payment systems before moving on to discuss two of the more significant protocols used for securing CNP transactions. In describing this model (also referred to as a pull model) a number of steps are necessary to complete a given transaction (see Fig. 1).

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, 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

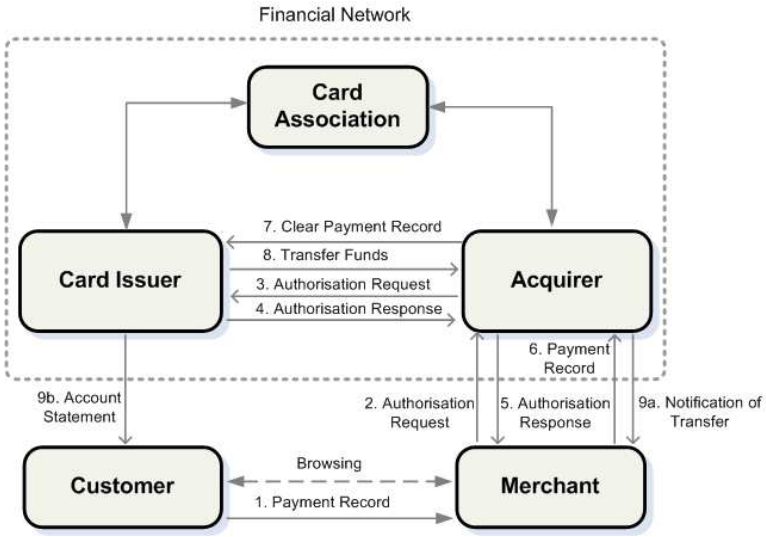

Figure 1: Generic model for card processing.

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.

2.1 Protocols for protecting CNP payments

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-D Secure: 3-D Secure and both Visa's [4, 5] and MasterCard's [16] 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. 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.

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 generates new transactions using the observed authorisation data. Such a script has been dubbed a *transaction generator* in [18]. Additionally, a phishing site that purports to provide a 3-D secure plug-in capability could potentially dupe cardholders into revealing authentication data.

3 Trusted Computing

Trusted Computing as discussed here, relates directly to the types of systems proposed by the Trusted Computing Group (TCG). The current documentation from the TCG encompasses a vast set of specifications ranging from Personal Computer (PC) [14] and server systems to (initial) specifications for trusted mobile platforms [30]. However, the specification set produced by the TCG is by no means the only work on Trusted Computing. Trusted Computing also encompasses new processor designs [9, 2] as well as OS support [24, 25].

However, it is the TCG's specifications for microcontroller design that have perhaps become most synonymous with Trusted Computing. The Trusted Platform Module (TPM) [31, 32, 33] forms the core of all efforts in Trusted Computing. The TPM itself comes in the form of a microcontroller with Cryptographic Co-processor (CCP) capabilities and resides on a platform's motherboard. The TPM is capable of providing the following functionality: a number of special purpose registers for recording platform state; a means of reporting this state to remote entities; secure volatile and non-volatile memory; random number generation; a SHA-1 hashing engine; and asymmetric key generation, encryption and digital signature capabilities.

In providing this functionality there are two particular cryptographic keys that have a special meaning. These keys are the Endorsement Key (EK) and the Attestation Identity Key (AIK). 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 state information.

However, in order for an AIK to have meaning outside of the confines of a particular platform, it is necessary for the platform to obtain a credential for an AIK from a trusted third party. How this credential is obtained differs between version 1.1b and version 1.2 of the TCG 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) whilst retaining the Privacy CA model for backward compatibility. For simplicity, for the remainder of this paper we will concentrate our discussion solely on the Privacy CA model.

Within this model, credential acquirement is achieved as follows: a Collate Identity Request command [15, 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 (signed by the TPM and platform manufacturer). Provided the evidence presented by a requestor's platform can be 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 [33, pp.151]. This then allows an AIK private component to be used to generate signatures over platform state information.

A recent addition to the concept of remote attestation has been the introduction of the Subject Key Attestation Evidence (SKAE) X.509 extension [13]. This extension provides a standard mechanism through which a verifying party can be assured of the security properties of a private key within a TPM. The security properties of a private key include both key type, which indicates whether a key is migratable or not, and attribute designation, which indicates what the key can be used for: signing, storage or both. After obtaining an AIK credential (following the method outlined above), a user signs the public component of either a non-migratable key pair (a key which is not allowed leave a TPM in an unencrypted form) or a Certified Migration Key pair [12] (CMK, a key which is allowed to leave a TPM but only under strict conditions). The signature on the public component is produced using the private component of an AIK. The user then applies to an SKAE CA for certification of the corresponding TPM-controlled (non-migratable or CMK) public key. If the CA is satisfied as to the AIK/public key binding, then a public-key certificate is issued by the CA to the platform. Here the certificate not only includes the public key which has been cryptographically bound to a TPM but also includes enough information for the relying party to validate this binding.

For the interested reader, introductory texts on Trusted Computing include [22, 23].

4 Application of Trusted Computing to CNP Transactions

In this section we will look at the issue of customer enrollment with a view to obtaining certification of a TPM-controlled (non-migratable or certified migratable) key. We present a number of different system architectures through

which enrollment may occur and discuss the issues of client-side certification in the face of the omnipresent threat of malware.

4.1 Enrollment

This section aims to explore different architectural options for enrolling a platform, and by extension its owner (cardholder), using a card-issuer-controlled Trusted Computing CA. The goal here is for a cardholder to obtain an X.509 certificate incorporating both card account details as well as a cardholder's public key, with the corresponding private key being bound to the cardholder's TPM. This certification by the card issuer will effectively bind a cardholder's hardware platform to a particular card. The cardholder can later demonstrate this binding when authenticating himself to a merchant during a CNP transaction. Thus the TPM acts as both a secure storage area for the cardholder's private key as well as providing a means by which the use of the private key can be controlled.

In order for a card issuer to provide an enrollment facility for their customers' platforms, it will be necessary for the card issuer to provide some form of CA functionality. Given the limitations of the TCG specifications, this functionality can take the form of either a Privacy CA, an SKAE CA or possibly both. As we saw in Section 3, in order for a platform to obtain an X.509 certificate for a TPM resident key it is necessary to go through a number of steps. A platform at the behest of its owner (the cardholder) first makes a request to a Privacy CA to certify an AIK public key. The corresponding AIK private key is then used to sign the public key of a non-migratable (or CMK) TPM key pair. This signed public key is then sent to an SKAE CA who certifies that the private portion satisfies certain key type and attribute designation constraints (as evidenced in the TPM_Certify_Info structure) before issuing an X.509 certificate on the non-migratable (public) key.

4.2 Deciding on an Architecture

Figure 2 shows the general certificate enrollment hierarchy in which customers can enroll with multiple card issuers who in turn can enroll with multiple card associations. The cardholders themselves have no direct dealing with the card association but instead interact with the enrollment interfaces exposed by their card issuers. In defining these interfaces there are various design decisions related to a card issuer providing Privacy/SKAE CA functionality. These can be broken down as follows:

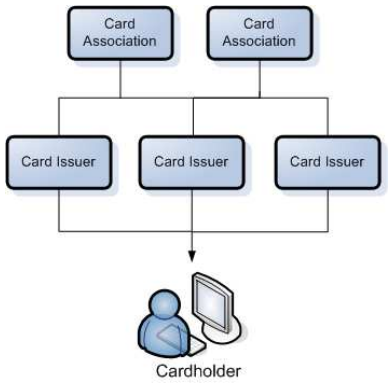

Figure 2: Certificate Enrollment Hierarchy

Privacy CA: By acting as a Privacy CA, a card issuer can issue AIK certificates to its customers' TPM-enabled platforms. Unfortunately, the usefulness of this approach is limited by the fact that an AIK is only allowed to sign integrity metrics and non-migratable/CMK keys, but not information generated outside a TPM.

Note that in discussing the use of a Privacy CA here, we are not suggesting that our approach should actually benefit from the privacy-enhancing features available from this choice. Rather, we see the card-issuer playing the role of a Privacy CA as providing a convenient mechanism for achieving client-side authentication within the limitations of the TCG specifications. Indeed, there is a potential privacy concern for customers in the disclosure of a platform's EK public component to a non-manufacturing entity: since an EK is unique per platform instance, it may act as a 'super-cookie' in identifying subsequent platform actions across multiple domains.

SKAE CA: By acting as an SKAE CA, a card issuer can issue X.509 certificates on non-migratable/CMK keys to customers' TPM-enabled platforms. Once this certificate is received it can be used in future transactions, either during an SSL handshake (see Section 5.1) or in support of a 3-D Secure authentication (see Section 5.2). In providing this service a customer's card issuer does not need to provide Privacy CA functionality. This can be provided by an entity that is in the best position to do so, typically a TPM manufacturer. However, a card issuer would need to trust the outcome of the Privacy CA AIK credential issuance procedure that precedes a customer's SKAE application. This solution can be seen to offer additional anonymity to a customer's platform, as it breaks the link between an EK and an AIK by having a Privacy CA (outside of the bank's domain) handle this mapping.

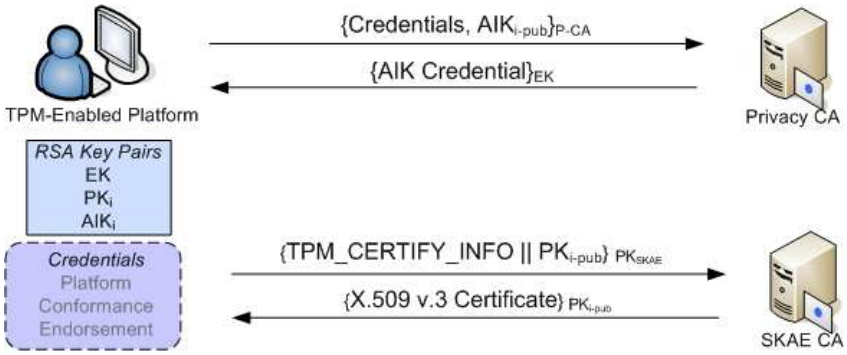

Figure 3: Card Issuer Controlled Privacy/SKAE CA

Hybrid CA: The final option is to have a card issuer act as a dual Privacy/SKAE CA. This is perhaps the most pragmatic solution for customer enrollment as it avoids the assumption that Privacy CAs are widely available. It also has the added benefit of shortening the customer enrollment procedure. Instead of making two separate CA requests, a customer generates an AIK and a non-migratable/CMK key pair and signs the public component of the non-migratable/CMK key using their private AIK key. The AIK/SKAE certificate request package is then bundled and sent to the Hybrid CA, which processes each component individually before issuing an AIK credential for the AIK and an X.509 certificate for the non-migratable/CMK key.

Figure 3 shows the most general case where a Privacy CA and SKAE CA are distinct entities. Obtaining a X.509 certificate for a TPM-bound non-migratable key in our system is a result of the following process:

1. The cardholder instructs his TPM to create an AIK key pair, AIK_{i-pub} and AIK_{i-priv} for the public and private components respectively.

2. The cardholder instructs his TPM to generate a certificate request package for their card issuer's Privacy CA in order to obtain an AIK credential for their newly generated AIK key, AIK_{i-pub}.

3. The Privacy CA validates the cardholder's request and issues an AIK Credential to the cardholder's TPM.

4. The cardholder's TPM receives the AIK Credential and the cardholder instructs his TPM to activate his AIK, AIK_{i-priv}.

5. The cardholder instructs his TPM to generate a key pair K_{i-pub} and K_{i-priv} with K_{i-priv} having the following properties: its type should

be non-migratable, its attribute designation should be signing only, and the use of the key should always require authorisation[3] which the cardholder now supplies.

6. The cardholder instructs his TPM to certify (sign) K_{i-pub} generated in Step 5 using AIK_{i-priv} generated in Step 1. This creates a signed TPM_CERTIFY_INFO structure [32] describing the security properties K_{i-priv} from Step 5.

7. The cardholder instructs his TPM to create an SKAE extension. This extension acts as a receptacle for a TPM_CERTIFY_INFO structure [32] from the preceding step.

8. The cardholder instructs his platform to create a certificate request package incorporating the SKAE extension from the previous step. During this process the cardholder authenticates himself to his card-issuer-controlled SKAE CA. This authentication would involve demonstrating knowledge of the payment card's PAN, CSC, address as well as a secret Personal Identification Number (PIN) or password[4].

9. If the card issuer SKAE CA is satisfied with the above information, then the SKAE CA issues an X.509 v.3 certificate containing a customer's PAN with an SKAE extension incorporating the K_{i-pub} of the non-migratable key pair generated in Step 5. The inclusion of the PAN in the certificate provides a mechanism through which a card can be demonstrably bound to a platform and by extension the platform's owner (cardholder). The omission of the CSC from the certificate removes certain security issues with respect to backward compatibility. Without a CSC/PAN combination, an adversary cannot engage in traditional MOTO-based payment authorisation. Thus the absence of the CSC from the X.509 certificate effectively neuters the value of the PAN to an adversary. Additionally, a subject can be identified using X.500 syntax which can be used directly in an AVS system (see Section 2). Finally, including a validity period can further constrain a card's usage, as is common in physical deployments.

10. The cardholder's platform receives the certificate from their issuing bank.

[3]We assume such authorisation data can be observed by malware. Its inclusion is to mitigate against the risk of this key later being misused in the event of a platform being stolen.

[4]We assume a secret PIN or password would be provided to cardholders using an out-of-band mechanism, similar to that currently used in Internet banking.

Whilst it may appear that the burden for a cardholder is exorbitant in the above process, in reality an application such as a card issuer supplied applet that interacts with a platform's TCG Software Stack could perform the majority of the cardholder's interactions with a TPM. The cardholder would only need to select and enter an authorisation string at Step 5 and a PIN/password at Step 9.

4.3 Client-Side Certification and Malware

The concept of client-side certification, as outlined in Section 4.1, works well if we assume an attack model that centers around external threats. However, as we have seen in Section 1, a model which only considers external threats is not always appropriate in CNP transactions. In order for a cardholder to generate a signature using the private component of the key referenced in the X.509 certificate, the cardholder needs to send authorisation data to their TPM to activate their signature key. However, this authorisation information may be observed and replayed by malware to generate new transactions [18]. Moreover, malware may be capable of modifying transaction data that is sent to the TPM for signing.

Our proposed mitigation for this malware problem is to use the TCG requirement that TPM-enabled platforms support a *secure attention sequence*, through which a user can demonstrate physical presence to a TPM. Here the design of a physical presence mechanism "should be difficult or impossible to spoof by rogue software" [31]. The combination of customer-provided card account details and evidence of the successful completion of a secure attention sequence can demonstrate that an authorised customer instigated a transaction. Malware on its own should be incapable of generating the required secure attention sequence.

The demonstration of physical presence on a TPM-enabled platform is typically associated with administrative functions of the TPM. However, physical presence may also be demonstrated ustilising the TPM_SetCapability and TPM_GetCapability commands [33]. These two commands can be used to set and retrieve bits in the Deferred Physical Presence Bit Map (DPPBM) that forms part of a TPM's TPM_STCLEAR_DATA structure [32].

In order for a cardholder (or more precisely an *untrusted* piece of software operating on a cardholder's behalf) to produce verifiable evidence of a (physical) commitment to a transaction, a cardholder needs to issue a series of commands to their TPM. A cardholder opens an *exclusive and logged transport session* [31] and calls the TPM_SetCapability to clear a single bit in the DPPBM. This command does not require a demonstration of physical presence and is used to prevent a bit from a previous transaction be-

ing reused by malicious software. Following this, a cardholder again calls TPM_SetCapability, but this time to set the newly cleared bit in the DPPBM (here the setting of the bit requires the cardholder to demonstrate physical presence). The cardholder next calls TPM_GetCapability to read the newly set bit indicating that physical presence has been demonstrated. Finally, a cardholder calls TPM_ReleaseTransportSigned to generate a *physical presence certificate*. The TPM_ReleaseTransportSigned produces a signature using K_{i-priv} over a data structure that includes a hash of the transport session log (consisting of the inputs, commands, and outputs encountered during the entire transport session) and a merchant-supplied anti-replay nonce. This nonce is constructed as a hash of the current transaction concatenated with a merchant-supplied random number. This physical presence certificate, together with the merchant's nonce and the transport session log, can be used to construct a *physical presence package* which a third party can verify. Note that, in order to load and use the key K_{i-priv}, the cardholder will need to input valid authorisation data. This is not intended to provide a defence against malware, but instead to prevent use of a stolen platform.

Unfortunately, user education now surfaces as a potential weak link in the security chain: malware may attempt to fool a user into providing a demonstration of physical presence. This is exacerbated by the fact that the manner in which physical presence functionality is presented to an end-user is entirely dependent on how a manufacturer chooses to implement it. Attesting to physical presence may be better suited to constrained devices such as mobile phones that conform to the Trusted Mobile specifications [30]. Here, the range of available mechanisms would be restricted by functional limitations of the mobile device. A second significant drawback is that the use of secure attention sequences will not prevent malware from modifying an on-going transaction (as opposed to generating multiple new transactions). Here we have to rely on the lack of a strong economic incentive for malware to behave in this way – we can assume that it will simply not be beneficial for malware to modify individual transactions, since this would lead to rapid detection for little benefit (from the malware's perspective). Stronger guarantees are unlikely to be available via Trusted Computing until recent initiatives such as Intel's La Grande [9] and AMD's Pacifica come to fruition. These efforts aim to provide additional Trusted Building Blocks (TBBs) in the form of hardware features, which can be exploited by next generation Operating Systems to provide strong security properties (such as non-interference, non-observation, and trusted I/O paths) for executing processes.

4.4 Migration

Our architecture allows a customer's private key that is bound to a particular TPM to migrate in a controlled manner to another TPM-enabled platform. We achieve this using the TPM's certifiable migration functionality. Certifiable migration is a TPM-specified operation [31] that permits the secure transfer of CMKs from one TPM-enabled platform to another. This transfer occurs in such a manner as to allow the new platform full usage of the migrated key. The process of migrating a CMK requires the inclusion of additional infrastructure components, in the form of either (or both) a Migration Selection Authority (MSA) and a Migration Authority (MA). A MSA is a third party responsible for approving the migration of a CMK to a specified destination (and does not directly handle the key itself). An MA preforms a similar function to that of an MSA, however, an MA will act as a temporary storage area for a CMK as the CMK transits to its ultimate destination.

In our architecture, irrespective of whether an an MSA or MA is used, it is important that the customer's card issuer is specified as the controlling entity when the key is created. Without specifying the card issuer as either the MA or MSA, malware would be capable of migrating the CMK directly to an untrusted platform controlled by a fraudster.

As the mechanism used to demonstrate physical presence is independent of CMK used to sign this demonstration, once the CMK has been migrated it can be used as described in Section 4.3 to generate the *physical presence package*.

5 Augmenting Existing Protocols with Trusted Computing

5.1 SSL Augmentation

SSL augmentation involves the addition of client (customer) authentication as provided (but seldom used) in standard implementations of SSL. Under the assumption of ubiquitous Trusted Platform Modules and the corresponding infrastructure that will be necessary to support them, we can use the enrollment mechanism outlined in Section 4 to provide a bootstrapping mechanism for providing client-side SSL certification.

The SSL process described here is identical to that of a standard SSL handshake in which client (cardholder) certification is requested by the server (merchant). Here the server requests a certificate and, if the client is in possession of an X.509 certificate that satisfies the merchant's request, then the

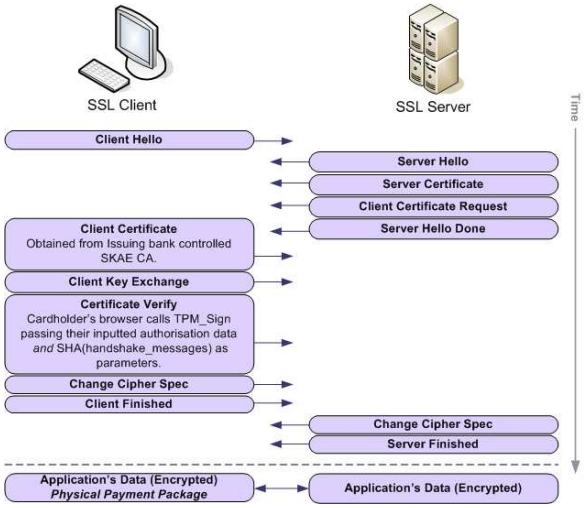

Figure 4: Augmenting SSL Authentication

cardholder's platform forwards this certificate to the merchant along with a certificate verify message. This certificate verify message provides a proof of possession for the private key, K_{i-priv}, corresponding to the public key, K_{i-pub}, referenced in the client certificate. Here a customer's TPM is responsible for performing customer authentication prior to using K_{i-priv} to generate the certificate verify message. The process of generating this certificate verify message requires the authorisation data for K_{i-priv} (as supplied by the cardholder) as well as all the handshake messages exchanged thus far. These two parameters are input to a TPM_Sign command [33]. This command checks to see if the provided authorisation data matches the authorisation data stored with the private component of the requested non-migratable key. If they match then the TPM uses the K_{i-priv} to generate a signature over the provided handshake messages. This signature is then passed to the merchant server for validation, subsequent to which the SSL handshake protocol proceeds as normal and the client can send their commitment to a transaction via the *physical presence package*.

The approach described above is an extension of that first described in [7] in the context of authentication in peer-to-peer networks.

5.2 3-D Secure Integration

As we reported in Section 1, there has been some movement recently in bringing unconnected card-readers (based on Mastercard's Chip Authentication Program (CAP) proposal) to market.

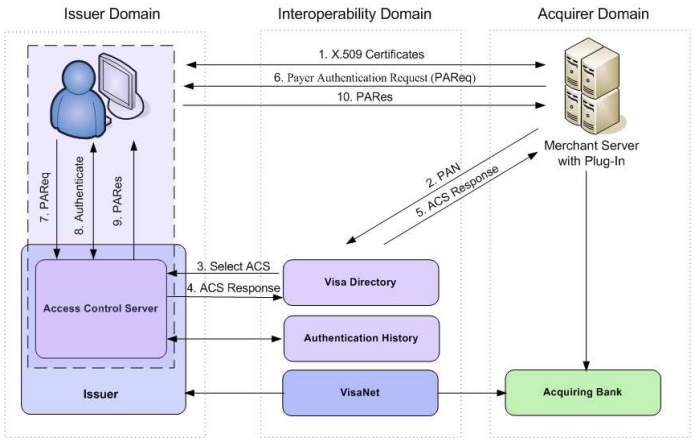

Figure 5: 3-D Secure Authentication

An alternative approach could be to use Trusted Platforms as a means of authorising transactions. Enrollment into a 3-D Secure-like environment would occur as laid out in Section 4. A TPM-enhanced 3-D Secure purchase transaction flow would proceed as follows:

Initiate Purchase: This stage is representative of the typical 3-D secure initiation procedure revolving around the merchant plug-in component, and can be seen in Steps 1-6 of Figure 5. Pursuant to a customer payment initiation request, the merchant plug-in contacts the Visa Directory Server which provides the address of an appropriate Access Control Server (ACS). The ACS response is then forwarded through the merchant plug-in and back to the customer's browser via a Payer Authentication Request (PAReq).

Payment Authentication: This stage is representative of the typical 3-D secure payment authentication process, except that the customer authentication mechanism in this setting involves the generation of the *physical presence package* outlined in Section 4.3, Step 8 of Figure 5. This approach allows the ACS server to be sure that a valid customer is proffering their valid account details as the customer's PAN forms part of the X.509 certificate and the customer has physically preformed an act that indicates approval of the transaction. Additionally, in this instance, the CSC, which is not included in the certificate, could effectively act as a PIN in further establishing a binding between a card its cardholder.

Payment Validation: Payment validation is a result of an examination, by the merchant plug-in, of the Payment Response message generated by

the ACS server. If the ACS signature validates correctly, then the merchant server can be assured that he is dealing with the valid owner of the presented payment card.

By using a 3-D secure authentication procedure that is augmented by Trusted Computing, we can achieve the benefits of an unconnected card reading facility without the need for additional client-side security tokens, under the assumption of TPM ubiquity. As we saw in Section 4.3, the demonstration of physical presence (Step 8 – payment authentication) partially combats the threat posed by malware by requiring the customer to perform a physical action as part of the payment authentication process.

The primary advantages of this approach over an approach based on unconnected card readers are its lower cost and its capability to support more flexible deployment. An unconnected card reader, once deployed, is a static device that cannot be updated without incurring the costs of reprovisioning every device. Using Trusted Computing allows a much finer-grained control over the life-cycle process. Moreover, the security afforded to a CNP transaction can take advantage of additional Trusted Building Blocks as and when they become more widely available in the marketplace.

6 Conclusions and Future Work

The use of the payment cards as an avenue for e-commerce is increasing at an unprecedented rate. In the physical world, the introduction of EMV for card-based payments at point of sale terminals has seen a dramatic reduction in the level of chargeback-related fraud. This is primarily due to the widespread tamper-resistent cryptographic hardware being deployed, preventing the cloning of cards.

Unfortunately, the benefits seen in the physical deployment of EMV for card payment transactions cannot be so easily gained in CNP scenarios. In this setting, knowledge of customer account information is all that is required to perform a transaction. This makes it impossible for a merchant or a customer's card issuer to determine if a valid owner of the account details being proffered is the one that actually instigated the transaction.

This paper has attempted to address this imbalance by analysing the role Trusted Computing can play in augmenting two different mechanisms for securing CNP transaction details. We showed how the integration of Trusted Computing physical presence signals with SSL and 3-D Secure can enhance the security of CNP transactions and partially address the threat posed by malware. In doing so, we highlighted how solutions based on 3-D Secure are

more amenable to the inclusion of physical presence signals since the merchant plug-in supplied by the financial network domain can be programmed to verify customer-supplied attestations of physical presence. With SSL, much greater heterogeneity of implementations of server logic are possible, since it is the merchant and not the financial network domain that decides on the actual implementation. By tying payment authorisations to Trusted Computing hardware, in the form of a TPM, we provide similar benefits to those obtained with EMV. That is to say, knowledge of a customer's account details is no longer sufficient to complete a transaction; rather, a customer would need to demonstrate possession of a private key which is physically bound to a piece of hardware under their direct control. This approach can be easily adapted to other payment protocols such as SET, or indeed any protocol where it is important that a human presence be determined.

As a natural extension of our work, we plan to examine how the extended security properties made available by potential future deployments of Trusted Computing technology (in which Trusted Computing functionality is fully integrated with OS and CPU) can be used to enhance the security of card payments. In particular, we propose to study the role that hardware-based virtualisation can play in protecting software-based EMV cards.

References

[1] A. Alsaid and C. J. Mitchell. Preventing phishing attacks using trusted computing technology. In *INC 2006: Sixth International Network Conference*, pages 221–228, July 2006.

[2] T. Alves and D. Felton. TrustZone: Integrated Hardware and Software Security — Enabling Trusted Computing in Embedded Systems. White paper, ARM, July 2004. `http://www.arm.com/pdfs/TZ\_Whitepaper.pdf`.

[3] APACS. Fraud – the facts 2007. `http://www.apacs.org.uk/resources_publications/documents/FraudtheFacts2007.pdf`, April 2007.

[4] Visa International Service Association. 3-D Secure™ Protocol Specification: Core Functions. `http://international.visa.com/fb/paytech/secure/main.jsp`, July 2002.

[5] Visa International Service Association. 3-D Secure™ Protocol Specification: System Overview. `http://international.visa.com/fb/paytech/secure/main.jsp`, May 2003.

[6] B. Balacheff, D. Chan, L. Chen, S. Pearson, and G. Proudler. Securing intelligent adjuncts using trusted computing platform technology. In *IFIP TC8/WG 8.8 4th Working Conference on Smart Card Research and Advanced Applications*, IFIP TC8/WG 8.8, pages 177–195, 2000.

[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] M. Bellare, J.A. Garay, R. Hauser, A. Herzberg, H. Krawczyk, M. Steiner, G. Tsudik, and M. Waidner. iKP – A Family of Secure Electronic Payment Protocols. In *WOEC'95: Proceedings of the 1st conference on USENIX Workshop on Electronic Commerce*, pages 89–106, Berkeley, CA, USA, 1995. USENIX Association.

[9] Intel Corporation. *LaGrande Technology Preliminary Architecture Specification*, May 2006. `http://www.intel.com/technology/security/`.

[10] S. Gajek, A.-R. Sadeghi, C. Stüble, and M. Winandy. Compartmented security for browsers - or how to thwart a phisher with trusted computing. pages 120–127, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[11] Trusted Computing Group. Trusted computing: Opportunities and challenges. `https://www.trustedcomputinggroup.org/downloads/tcgpresentations/`, 2004.

[12] Trusted Computing Group. *Interoperability Specification for Backup and Migration Services*, revision 1.0 edition, 2005.

[13] Trusted Computing Group. *TCG Infrastructure Workgroup Subject Key Attestation Evidence Extension*, 1.0 edition, June 2005.

[14] Trusted Computing Group. *TCG PC Client Specific Implementation Specification For Conventional BIOS*, 1.2 final edition, 2005.

[15] Trusted Computing Group. *TCG Software Stack Specificiation Version 1.2 Level 1*, 2006.

[16] MasterCard International. SecureCode™ Merchant Implementation Guide. `http://www.mastercardmerchant.com/securecode/`, March 2004.

[17] C. Jackson, D. Boneh, and J. Mitchell. Spyware resistant web authentication using virtual machines. `http://crypto.stanford.edu/antiphishing/spyblock.pdf`.

[18] C. Jackson, D. Boneh, and J. Mitchell. Transaction generators: Rootkits for the web. In *Proceedings of the 2nd USENIX Workshop on Hot Topics in Security (HotSec 2007)*, August 2007.

[19] B. Krebs. Citibank phish spoofs 2-factor authentication. `http://blog.washingtonpost.com/securityfix/2006/07/citibank\_phish\_spoofs\_2factor\_1.html`, 2006.

[20] J.M. McCune, A. Perrig, and M.K. Reiter. Bump in the Ether: A Framework for Securing Sensitive User Input. In *Proceedings of the 2006 USENIX Annual Technical Conference*, pages 185–198, June 2006.

[21] P. Meadowcroft. Combating card fraud. `http://www.scmagazine.com/uk/news/article/459478/combating+card+fraud/`, January 2005.

[22] C.J. Mitchell, editor. *Trusted Computing*. IEE Professional Applications of Computing Series 6. The Institute of Electrical Engineers (IEE), London, UK, April 2005.

[23] S. Pearson, editor. *Trusted Computing Platforms: TCPA Technology in Context*. Prentice Hall, Upper Saddle River, New Jersey, USA, 2003.

[24] M. Peinado, Y. Chen, P. England, and J. Manferdelli. NGSCB: A Trusted Open System. In H. Wang, J. Pieprzyk, and V. Varadharajan, editors, *Proceedings of 9th Australasian Conference on Information Security and Privacy, (ACISP '04)*, volume 3108 of *Lecture Notes in Computer Science (LNCS)*, pages 86–97, Sydney, Austrailia, 13–15 July 2004. Springer–Verlag, Belin–Heidelberg, Germany.

[25] M. Peinado, P. England, and Y. Chen. An Overview of NGSCB. In C. J. Mitchell, editor, *Trusted Computing*, IEE Professional Applications of Computing Series 6, chapter 7, pages 115–141. The Institute of Electrical Engineers (IEE), London, UK, April 2005.

[26] L.F.G. Sarmenta, M. van Dijk, C.W. O'Donnell, J. Rhodes, and S. Devadas. Virtual monotonic counters and count-limited objects using a tpm without a trusted os. In *STC '06: Proceedings of the first ACM workshop on Scalable trusted computing*, pages 27–42, New York, NY, USA, 2006. ACM Press.

[27] SETCo. SET Secure Electronic Transaction 1.0 specification — the formal protocol definition. http://www.setco.org/set_specifications.html, May 1997.

[28] A. Spalka, A.B. Cremers, and H. Langweg. Protecting the creation of digital signatures with trusted computing platform technology against attacks by trojan horse programs. In *Proceedings of the IFIP SEC 2001*, pages 403–420, 2001.

[29] Symantec. Symantec Internet Security Threat Report Volume XI. Available on-line, March 2007. http://www.symantec.com/enterprise/theme.jsp?themeid=threatreport.

[30] TCG. *The TCG Mobile Trusted Module Specification*, 0.9 revision 1 edition, 2006.

[31] TCG. *TPM Main: Part 1 Design Principles*, 1.2 revision 103 edition, 2007.

[32] TCG. *TPM Main: Part 2 Structures of the TPM*, 1.2 revision 103 edition, 2007.

[33] TCG. *TPM Main: Part 3 Commands*, 1.2 revision 103 edition, 2007.