Don't trust POS terminals! Verify in-shop payments with your phone

Iulia Ion* and Boris Dragovic

CREATE-NET, Trento, Italy, *ETH Zurich, Switzerland
iulia.ion@inf.ethz.ch, boris.dragovic@gmail.com

Abstract. Despite the advanced capabilities of the chip-enabled, debit and credit cards, fraud in payment transactions has not diminished - it has shifted. The reason lies in the lack of a trusted communication path between the smartcard and the cardholder. More explicitly, because users have no means of verifying the authenticity of the Point-of-Sales (POS) terminal, they do not know how much they are about to pay nor to whom. We propose to use the camera-enabled mobile phone and a previously shared secret to create a two-way, secure communication channel. Messages from the card are displayed by the POS terminal as visual codes, then captured and decoded by the phone. Messages from the cardholder are computed by the phone and manually typed in as one-time PINs. We extend the EMV payment protocol to provide explicit verification and confirmation of the transaction amount. In the process we also improve cardholder authentication, protect against stolen PIN and cards, and eliminate the POS terminal from the trust chain altogether. The implementation requires minimal software updates and no hardware modifications.

1 Introduction

Security-wise, credit and debit card technology has evolved from rather insecure, magnetic strip based cards to, admittedly more secure, Integrated Chip Card (ICC). Most notable attacks on the former rely on the ease of cloning the magnetic stripe, visual signature verification as well as the supported complete lack of authentication in the extreme cases. Unfortunately, the ICC technology does not come without its woes either. Despite the introduction of EMV [11], (named after Europay, Mastercard and Visa, also known as Chip and PIN) which should have made systems more secure, the payment fraud did not diminish - it shifted focus.

In this paper we address a threat stemming from the inclusion of the Point-of-Sale (POS) terminal in the ICC payment trust-chain. In other words, we address the issue of frauds arising from tampered POS terminals overcharging customers. In the typical scenario, the customer authorizes the payment (by typing in the PIN) based on the amount displayed on the POS terminal built-in display. On the other hand, the card signs the payment transaction corresponding to the

amount reported to it by the terminal. Unfortunately, these two amounts can be different - should the POS terminal have been tampered with.

In a simple scenario, Alice thinks she is paying 50 euros for a book and she approves the transaction by entering the PIN, but the card is requested to produce a Transaction Certificate (TC) for 100 euros. A number of days can pass before Alice spots the fraud on her bank statement. This, unfortunately, may happen even in the presence of SMS notification due to a number of reasons, e.g. offline terminal, delayed transaction processing, lack of network coverage.

The shortcoming of the current POS terminal based payment schemes is the lack of a *direct* trusted communication path between the chip on the card and the cardholder. Thus, the cardholder has absolutely no means of verifying the transaction amount real-time should the terminal be compromised. Despite their enhanced processing power and capabilities, smart-cards cannot protect users against such frauds due to the fragmentation of the trust-chain.

Although manufacturers claim that POS terminals are designed to be tamper resistant and prevent unauthorized parties from altering the software or hardware, experiments proved differently. By replacing internal parts, Drimer and Murdoch were able to gain complete control over the terminal [9]. Furthermore, common users are unexperienced, not trained to recognize POS models nor to look for tampering signs. Even for experienced users, it is difficult to know all the approved POS terminal models on the market and to detect tampering.

In fact, in 2006, following the discovery of a systematic fraud that resulted in the theft of one million Pounds from customer accounts, Shell stopped accepting Chip and PIN payments at 600 out of 1,000 UK petrol stations [16]. Police suspected that fraudsters have altered the PIN pads or that employees were involved in the scam.

The contribution of this article is an extension to the payment protocol which effectively shortens the trust chain by excluding the POS terminal from it. Thus it closes the outlined class of frauds focusing on tampering with POS terminals.

2 Defending against credit card fraud

Several solutions have been developed to address the problem of compromised POS terminals. This section gives a brief overview and analyzes the most notable ones: transaction confirmation via SMS, trusted mediating hardware called *electronic attorney* and credit card technology that integrates display. We highlight the latency, cost and usability shortcomings of the solutions, motivating the contribution of this paper.

2.1 Alternative solutions

SMS confirmation. Many banks offer various flavors of SMS card transfer notification services where cardholders are made aware of the involvement of their card in a transaction. These are not confirmation services as the cardholders are given no opportunity to block the reported transaction.

A number of issues may impede prompt SMS delivery, e.g. network latency, lack of network coverage, delay in transaction processing as is the case in off-line POS terminals, etc. Therefore, not integrating SMS as a confirmation channel in the actual payment process also has external motivation. In this paper, we propose to actively involve the mobile phone in the payment transaction as a trusted, authorized instrument.

Electronic attorney. Anderson and Bond [2] propose the use of a trusted piece of hardware to mediate the physical interaction between the chip reader slot of the POS terminal and the credit card. The electronic attorney is "a small device about the size of a credit card, with chipcard contacts at one end and a chipcard reader at the other, as well as an LCD display and several buttons" [2]. The cardholder inserts his secret PIN on the trusted keyboard, thus preventing PIN sniffing, and verifies the charged amount on the display of the electronic attorney, thus preventing merchant fraud.

There are a number of obstacles for market success of such a product ranging from increased cost for the end-user, inconvenience of carrying yet another token, the need for substantial software updates to POS terminal, including fundamental modifications to the payment protocol itself. The mobile phone, on the other hand, represents a token that can be found in almost any pocket and represents the most used computing platform.

Display equipped cards. Although IC cards with integrated display could provide a viable solution to the problem, technology implementing it [4] is still very immature and expensive. In the future, slim credit cards with incorporated screens could display the approved transaction amount. That would, of course, require replacement of credit cards on a large scale - incurring costs that would probably be pushed to the end-user as is the case in biometric identification cards and passports. While users are not security aware and do not recognize the vulnerabilities in the payment protocol, banks do not have enough incentives to invest in customer's security [2] due to risk vs. investment asymmetry.

Unlike display equipped cards and the electronic attorney, our solution is much easier to implement, is cost-effective both for banks and customers, requires minimum implementation effort, and incurs no extra hardware costs.

2.2 Proposed solution: the mobile phone as a trusted component

To create a trusted communication path between the card and the cardholder, we make use of the customer's personal mobile phone and *visual code technology*[1]. Mobile phones are now the most widely deployed computing platform in the world. Analysts predict that one billion mobile phones will be sold in 2009 [15]. As highly personal devices, with enhanced computing capabilities, great connectivity and equipped with radio, mp3 player, video cameras, they represent a great platform for a big range of privacy and security applications, from entertainment, personal content management to identification tokens and payment

[1] 2d code: http://2d-code.co.uk/

systems. Subjective perception of mobile personal devices, such as phones, suggests that users regard them as intrinsically trusted.

Visual codes or *2D bar-codes*, representing the evolution of ubiquitous barcodes, are already widely used to encode information. These 2D codes are particularly suitable for being scanned and decoded using camera-enabled mobile phones. Applications range from hiding private data, annotating physical objects, to easily retrieving related information and functionality - application that has proven to be very popular in Asia, etc. Visual codes generation and recognition software is now widely available [8, 19]. In fact, in Japan most of the phones already come with preinstalled recognition software. With the use of

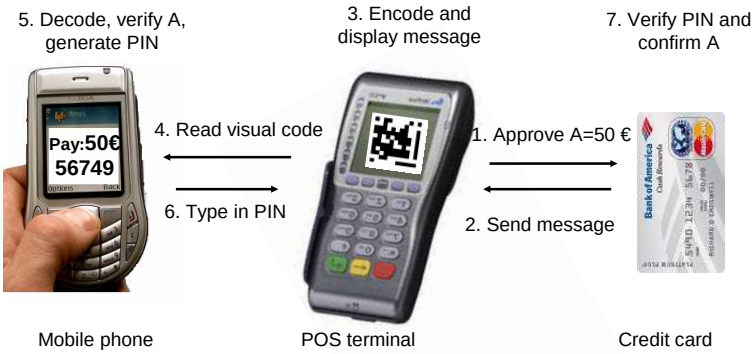

Fig. 1. EMV payment protocol

visual codes and a camera phone, the payment scenario proceeds as follows. The credit card and the personal mobile device share a secret key. Through the use of cryptographic primitives, the mobile phone acts as a trusted party and assists the cardholder in the payment process. Just like in the current payment scenario, (1)the cardholder inserts his credit card into the POS terminal. Before the user enters the PIN, the terminal communicates to the card the transaction amount. (2)The card produces a message based on the payment amount and a one-time challenge generated on the spot by the card. (3)The terminal encodes and displays the message in the form of a visual code on its display (where previously the amount to be verified was shown). (4)The cardholder then uses his mobile phone to scan and decode the visual tag automatically. The user can verify on the trusted display of his mobile phone the transaction amount. (5)Based on the one-time challenge computed by the card and on the pre-shared secret key (between the card and the phone), the mobile phone computes a one-time PIN. (6)If the amount coincides with the user's expectations, he enters the PIN into the payment terminal. (7)Being in possession of the secret key and the one-time challenge from step 2, the card itself verifies the PIN which confirms that the user agreed to the specific amount and it finally authorizes the payment transaction.

3 EMV payment protocol

In this section, we briefly detail the standardized transaction processing steps, as defined by the EMV specifications [12, 13], and executed between the card and the terminal. We start by introducing the involved components. The credit card and the POS terminal communicate by exchanging Application Protocol Data Units (APDU) = command-response message pairs, via the T=1 protocol.

The EMV payment steps differ, depending on the different types of card authentication employed: Static Data Authentication (SDA) (based on symmetric key algorithms), Dynamic Data Authentication (DDA) (cards with public key capabilities), and Combined Data Authentication (CCA) (symmetric key cryptography, the use of certificates and signatures). As the *Chip and SPIN* investigations states [3], there is no public, official data, neither from APACS[2] nor any UK bank, confirming whether DDA or SDA capable cards are being used in practice. However, after having analyzed a number of UK Chip and PIN cards, the authors have found no evidence to suggest dynamic capabilities.

Smart-cards are capable of securely storing several keys and certificates and of performing cryptographic computations such as hashing (SHA-1 algorithm), symmetric encryption (AES, DES, 3DES), public key cryptography (RSA) and digital signatures (DSA). The specific capabilities and algorithms depend on each type of card and manufacturer. Furthermore, several payment applications can be installed and run on one smart-card. Cryptographic keys are stored and computed securely on the card and never leave it.

Point of Sales terminals vary in brand name (Hypercom, VeriFone, IBM), dimensions, design, capabilities and price (from $100 to $800). They usually contain a card reader slot, Internet connection, small keyboard and small display, from usually 128x64 pixels simple ASCII capable display to the more sophisticated, programmable devices, e.g. to include the company logo and graphics as well as customized payment services.

Figure 2 depicts the transaction steps. The main steps are the following:

1. Read application data: When the card is powered up, it responds with an Answer to Reset (ATR) message, which contains information about the card. Over a series of SELECT and READ RECORD commands, the terminal requests relevant information to process the transaction, such as account details, supported payment applications and cardholder verification methods (CVM) to agree on a common payment scheme [12]. To easily select the payment application, the ICC maintains a directory structure for the list of installed applications, each of which is associated with an Application Identifier (AID).

2. Card authentication: This step is also called Internal Authentication. Basically, the card proves it is genuine to the external world. The terminal confirms the legitimacy of the card. Depending on the authentication protocol, SDA, DDA or CCA be used.

[2] APACS - the UK Payments Association: http://www.apacs.org.uk/

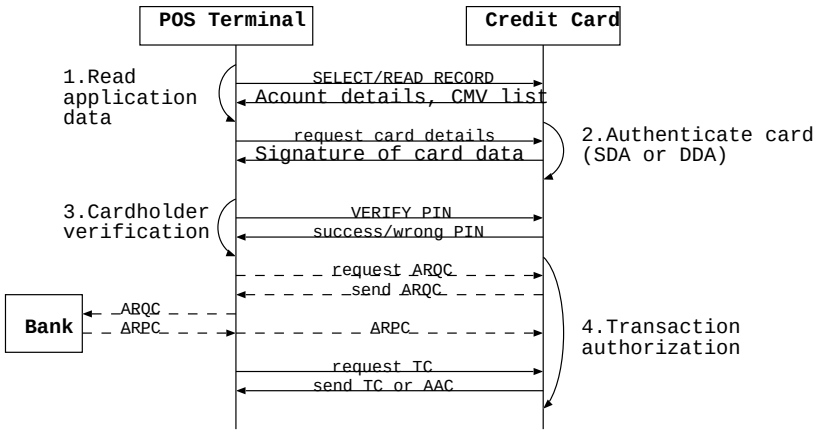

Fig. 2. EMV payment protocol

3. Cardholder verification: The user enters the PIN in the POS keyboard. The POS terminal sends the PIN to the card, which verifies it and confirms that the customer is the legitimate card owner (Offline PIN Processing [13], p.105)

4. Transaction authorization: Step 4 in Figure 2 depicts the transaction authorization process. The dash lines represent optional message exchanges (also called External Authentication); they only take place in the case of on-line transactions (i.e. if there is connectivity between the terminal and the issuing bank and the card requests authorization from the bank). The terminal requests a Application ReQuest Cryptogram (ARQC) - which contains the card, terminal, and transaction data encrypted by a DES key. The ARQC is then forwarded to the bank. As a result, the bank responds with a Application ResPonse Cryptogram (ARPQ) by which it authenticates itself and confirms that sufficient funds are available. The card then issues the Transaction Certificate (TC) which is later used by the merchant to collect the charged amount or denies the payment through an Application Authentication Cryptogram (AAC).

To improve efficiency of credit card payments, transactions are often not sent one-by-one to the bank as soon as they are collected by the terminal as a set of TCs. Such batching can be leveraged for lowering resource consumption (time for processing and communication, network costs, etc.) or may occur naturally in the case of off-line POS terminals (e.g. buying tickets in trains, food in the street or duty free goods in airplanes).

4 Extended payment protocol

To support the proposed extension to the EMV protocol presented above, several alterations are needed. As Figure 3 shows, these modifications are actually minimal; they influence the cardholder verification process and make the param-

eters for transaction authorization dependent on the results of the previous step. Steps 1 and 2 in the EMV protocol remain completely unchanged.

To be on the safe and robust side, we assume the simplest, most common case and design our protocol for off-line POS terminal and SDA card. By limiting the requirements on the system and card capabilities, we make the proposed EMV extension applicable to all types of transactions, even on cards with greater capabilities (e.g. DDA with on-line authentication).

4.1 Protocol steps

In the following specification of our extension to the EMV protocol, by N we denote a nonce (20 bits long, about one million possible values), A is the transaction amount (16 bits long, payments up to 65000 units, enough for Euro, Dollar, British Pounds transactions), K is the shared secret key (112 bits), $hash_K(m)$ is the secure hash of the message m (hash size of 56 bits) computed with the key K and $A||N$ is the concatenation of A and N.

1. The card generates a fresh N and computes $hash_K(A||N)$, then sends the terminal the message $m1{=}A||N||hash_K(A||N)$.
2. The terminal displays a visual code encoding the message $m1$.
3. The phone decodes the visual image and verifies the message authenticity by computing the hash with the shared secret key K.
4. The phone computes the response $m2{=}hash_K(N||hash_K(A||N))$ and trims it to 20 bits (i.e. 6-digit number), which is displayed to the user.
5. The user types the code into the payment terminal and the card verifies the inserted PIN by making the same computation. Afterwards, the card approves the payment transaction for the amount A.

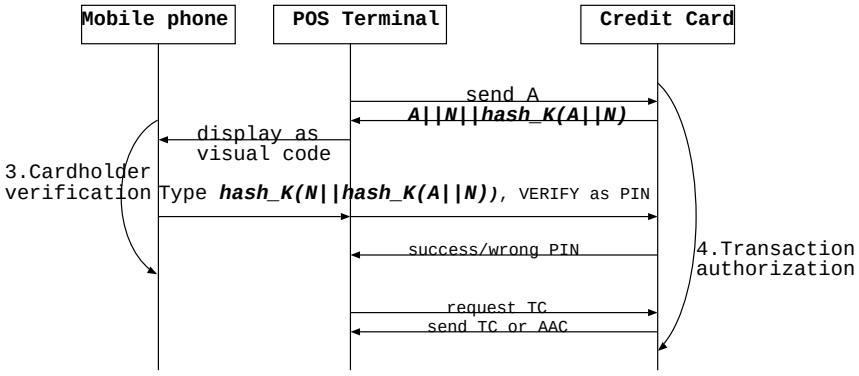

Fig. 3. Extended EMV payment protocol, steps 3 and 4

We trim message $m2$ to just 20 bits because the user would not quite enjoy typing in the entire hash. Even if done naively, the truncation of the hash to

obtain the PIN would remain secure for the following two reasons: i) the secret key used for hash computation is unknown to the terminal while the one-time challenge changes on each transaction; ii) the number of PIN entry attempts is limited while there is 1 in a million probability of guessing the PIN correctly. A similar approach is chosen by Wong and Stajano to design a multichannel security protocol in [21]. Making *m2* dependent on *A* provides resistance to replay attacks in the very unlikely case of collision in the values of the nonce *N*.

4.2 Implementation and installation

Smart-cards and POS terminals currently in use offer all the computation capabilities required for implementation of the suggested EMV protocol extension. Therefore, no hardware upgrade is needed on either side. Furthermore, updates to POS terminal firmware and card software can be done remotely, by authorized parties through standard interfaces. In fact, brand manufacturers periodically push new software updates into the POS terminals, and banks update card applications and risk management data via payment terminals. Furthermore, the card and the terminal can exchange data messages of arbitrary sizes. Following the fixed header formats, the byte denoted L specifies the length of the message that follows, which is interpreted by the selected payment application. For backward compatibility, we envisage support for both alternatives. The user might prefer the old EMV protocol if, for example, he or she forgot the phone at home or if his mobile device is temporarily out of battery power.

The shared key K shall be securely stored in a key file on the smart-card and in the protected storage of the mobile phone's SIM card, available only upon successful entering of a secret PIN, thus preventing unauthorized use in the case of device abduction.

The visual code should be large enough to encode 90 bits of data and compact to fit on the POS display. For visual codes, we can choose between two main formats. While QR Codes [1] have higher error correction capacity, Data Matrices are more compact [20]. Considering the limited displays of POS terminals and the physical proximity of the mobile phone, we recommend the use of Data Matrix. According to Ballagas et. all [6], a 10x10 visual code with error detection, can encode up to 76 bits of data. We propose increasing the size of the visual code to reach the desired encoding capacity.

The hashing algorithm employed should be implemented by EMV compliant cards. For secure messaging SHA-1 is used [12], whereas for Application Cryptograms (AC) 3DES MACs are generated[3]. Specific implementation details depends on manufacturer and are not included in the EMV technical specifications. For example, the estimated duration for performing 3DES encryption/decryption is 30ms for MPCOS-EMV, Gemplus cards [14]. Even though we were not able to find specific information on hashing performance, we do not expect the extra processing time to significantly influence the payment transaction

[3] Steven J. Murdoch, EMV flaws and fixes: vulnerabilities in smart card payment systems. COSIC Seminar, K.U. Leuven, Belgium, 11 June 2007. http://www.cl.cam.ac.uk/ sjm217/talks/leuven07emv.pdf

duration. In line with the current implementation, we propose the use of 3DES based hashing algorithms in the protocol extension.

5 Evaluation and discussion

It has been argued that mobile phones are, similarly to personal computers, susceptible to viruses and should, therefore, not be considered trusted environments. Furthermore, some argue that, due to software vulnerabilities, mobile devices are not secure enough to act as payment devices, as in the case of payments conducted with Near Field Communication (NFC) phones. However, by keeping the two-factor payment entities (credit card and mobile phone), we only increase the security of the EMV payment protocol, which currently exclusively relies on the computation capabilities of the smart-card. In our design, we were inspired by the new computing paradigm proposed by Balfanz et al [5]: improve security by migrating part of a PC application to a small, trusted device. The mobile device acts as a convenient, secure smart card.

Our contribution brings two-fold improvements to the protocol. Firstly, it enhances cardholder authentication through the use of one-time PIN, a uniquely generated PIN for every transaction. Two-factor authentication in on-line banking is offered by many banks as a value added service (such as Protiva from Gemalto[18]). Secondly, and most importantly, it protects against merchant fraud by supporting explicit confirmation of the amount to be charged for the payment transaction, for which simple One Time Password (OTP) schemes are not sufficient. While similar solutions as the one we have presented in this paper have been implemented for on-line banking [7], no prototype nor specific proposal has been done for in-shop payments.

The same system shortage that we address is pointed out by Drimer and Murdoch and exploited in a relay attack [10]. However, while performing a relay attack requires a counterfeit terminal, a counterfeit card and synchronization of two distant payments, we address a much easier to fulfill scenario, which can be carried out with just one crocked terminal. Furthermore, our solution also provides a stronger alternative to the distance bounding based solution proposed by the authors.

6 Conclusion and future work

In this article, we proposed the use of the camera-equipped mobile phone during in-shop credit card payments to create secure communication channels with the card. Through the use of visual codes and one-time PINs, we create explicit amount verification and confirmation and thus protect against merchant fraud. The proposed enhancement benefits not only the cardholder through protection against theft of PIN or credit card, but also the bank and the retailer through improved cardholder authentication.

In the future, we plan to implement a prototype of the application and conduct user studies. Because real POS terminals and credit cards are not reprogrammable by unauthorized parties, we plan to implement a prototype using Java Card Development Kitand/or smart-card readers (as credit cards) and the freely available OpenCard Framework[17] (as POS terminals). Support for cryptographic operations on mobile phones is already widely available: Java Card Applet, SATSA optional libraries, lite Bouncy Castle implementation.

References

1. Information Technology. Automatic Identification and Data Capture Techniques. QR Code 2005 Bar Code Symbology, ISO/IEC 18004. Technical report, International Organization for Standardization, 2006.
2. Ross Anderson and Mike Bond. The man-in-the-middle defence. In *Security Protocols Workshop*, March 2006.
3. Ross Anderson, Mike Bond, and Steven J. Murdoch. Chip and SPIN. http://www.chipandspin.co.uk/spin.pdf.
4. AVESO. Display enabled smart cards. http://www.avesodisplays.com.
5. Dirk Balfanz and Ed Felten. Hand-held computers can be better smart card. In *Proceedings of USENIX Security '99. Washington, DC.*, August 1999.
6. Rafael Ballagas, Michael Rohs, Jennifer G. Sheridan, and Jan Borchers. Sweep and point & shoot: Phonecam-based interactions for large public displays. In *CHI '05: Extended abstracts of the 2005 conference on Human factors and computing systems*, Portland, Oregon, USA, April 2005. ACM Press.
7. Cronto mobile client. http://www.cronto.com.
8. Diego López de Ipiña, Paulo R. S. Mendonça, and Andy Hopper. TRIP: A Low-Cost Vision-Based Location System for Ubiquitous Computing. *Personal and Ubiquitous Computing*, 6(3):206–219, 2002.
9. Saar Drimer. Chip & PIN terminal playing Tetris. http://www.lightbluetouchpaper.org/2006/12/24/chip-pin-terminal-playing-tetris/, December 24th, 2006.
10. Saar Drimer and Steven J. Murdoch. Keep your enemies close: Distance bounding against smartcard relay attacks. In *USENIX Security Symposium*, August 2007.
11. EMVCo. http://www.emvco.com/.
12. EMVCo. Book 1 - Application independent ICC to Terminal Interface requirements, 4.1 edition, May 2004.
13. EMVCo. Book 3 - Application Specifications, 4.1 edition, May 2004.
14. GEMPLUS. *MPCOS-EMV R4, Reference Manual*, November 2000.
15. BBC News. Mobiles head for sales milestone. http://news.bbc.co.uk/2/hi/technology/4697405.stm, 19 July 2005.
16. BBC News. Petrol firm suspends chip-and-pin. http://news.bbc.co.uk/1/hi/england/4980190.stm, 6 May 2006.
17. OpenCard Framework. http://www.opencard.org/.
18. Protiva. http://www.protiva.gemalto.com.
19. Michael Rohs. Real-world interaction with camera-phones. In *2nd International Symposium on Ubiquitous Computing Systems (UCS 2004)*, Tokyo, Japan, 2004.
20. Semacode. Choosing the best 2d barcode format for mobile apps. Technical report, 2006.
21. Ford Long Wong and Frank Stajano. Multichannel security protocols. *IEEE Pervasive Computing*, 6(4):31–39, 2007.