Optimised to Fail: Card Readers for Online Banking

Saar Drimer, Steven J. Murdoch, and Ross Anderson

Computer Laboratory, University of Cambridge, UK
http://www.cl.cam.ac.uk/users/{sd410,sjm217,rja14}

Abstract. The Chip Authentication Programme (CAP) has been introduced by banks in Europe to deal with the soaring losses due to online banking fraud. A handheld reader is used together with the customer's debit card to generate one-time codes for both login and transaction authentication. The CAP protocol is not public, and was rolled out without any public scrutiny. We reverse engineered the UK variant of card readers and smart cards and here provide the first public description of the protocol. We found numerous weaknesses that are due to design errors such as reusing authentication tokens, overloading data semantics, and failing to ensure freshness of responses. The overall strategic error was excessive optimisation. There are also policy implications. The move from signature to PIN for authorising point-of-sale transactions shifted liability from banks to customers; CAP introduces the same problem for online banking. It may also expose customers to physical harm.

Keywords: banking security, reverse engineering, authentication, liability, chip and PIN.

1 Introduction

The late Roger Needham once remarked that 'optimisation is the process of taking something that works and replacing it with something that almost works, but is cheaper'. The history of cryptographic protocols – both in the research literature and in the field – is littered with examples of optimisation; of protocols that failed because designers had left out some contextual or other information that, on casual inspection, had seemed unimportant but whose absence led to catastrophic failure. Anderson and Needham thus argued that in the protocol world, robustness is closely tied to explicitness [1]. This paper presents a new and disturbing real-world example of an actually deployed banking protocol that fails because it has been excessively optimised.

Online banking is growing almost everywhere; in the UK, for example, there has been a 174% increase in the number of users between 2001 and 2007 [2]. This is easy enough to explain: online banking is convenient for customers, and lets bankers cut their staff costs. But, as banking has moved online, fraud has followed. Losses in the UK from online banking fraud were £21.4m in the period

R. Dingledine and P. Golle (Eds.): FC 2009, LNCS 5628, pp. 184–200, 2009.

Fig. 1. NatWest and Barclays issued CAP readers on the left and right, respectively. An opened NatWest CAP is shown in the centre. These readers are given to bank customers for free.

of January to June 2008, an increase of 185% when compared to the same period of the previous year [3].

One of the most common forms of attack is "phishing". Here, criminals send emails impersonating banks, asking customers to click on a link under some false pretence; if they do, a malicious copy of their bank's website asks for their authentication data. Another common attack involves malware; authentication details are stolen by a software keylogger on the customer's PC.

To resist these attacks, some bank websites only ask for some characters from the customer's password, or ask for them to be entered in drop-down boxes rather than at the keyboard; some banks have switched to one-time passwords such as the printed "iTAN" codes used by German banks [4], or electronic one-time-password generators such as the RSA SecurID.

However, one-time passwords are still vulnerable to a real-time man-in-the-middle attack. Here, the malware or phishing website initiates a fraudulent transaction with the customer's bank at the same time as it prompts the customer for their password or one-time code. The process may even be triggered when the customer attempts a transaction, rather than prompting them to do one. In any event, as the fraudulent transaction is being performed at the same time as the customer is trying to do a real one, a time-dependent or one-time password will still be valid.

This class of attack can be resisted by cryptographically binding the one-time code to the data of the transaction being attempted – *transaction authentication*. A robust way to do this is to provide the customer with an electronic signature device with a trustworthy display on which she could verify the transaction data, a trusted path to authorise a digital signature, and a tamper-resistant store for the signing key.

Such devices were foreseen by the EU Electronic Signature Directive which provided for signatures thus created to be admissible as evidence in legal

Fig. 2. We used FPGA boards to snoop on CAP transactions (left) and emulate a card (right). Using a USB card reader we emulated a CAP reader (centre).

proceedings. However such devices typically cost $100 or more. The Chip Authentication Programme (CAP)[1] is a lower-cost implementation of this general approach.

Individual countries have adopted different variants of CAP based on the original specification. In this paper we examine the UK version. It uses the deployed "Chip & PIN" smart card infrastructure. Participating banks have sent out handheld smart card readers, shown in Figure 1, with keypads and displays which, with a customer's card and PIN, generate one-time passwords.

Even though Chip & PIN is based on the public EMV standard (named after its initiators – Europay, MasterCard, and Visa), the CAP standard is secret and so not subject to scrutiny, despite being a critical security component the public must rely on for banking transactions. Therefore, in Section 2 we describe the results of successfully reverse engineering the system. In Section 3 we describe how CAP is used in online banking, and in Section 4 a number of security vulnerabilities we discovered in the underlying protocol and its implementation by two British banks. Finally, we propose some improvements to the system in Section 5 and discuss policy implications of the failures in Section 6.

2 Protocol Description

We used three different techniques to reverse engineer the protocol. First, we monitored communications between legitimate cards and readers (Figure 2 left), using an FPGA based protocol analyser we designed. Second, we emulated a reader and challenged the card (Figure 2 centre). Finally, we constructed an FPGA based card emulator in order to interrogate the reader (Figure 2 right). In all three cases we fully controlled the input, at either the electrical interface or keypad, so our approach was in effect an adaptive chosen text attack. We did not attempt to extract or study the code running on either the smart card or CAP reader, so we cannot be certain that we have a full implementation of the

[1] CAP is the MasterCard brand; Visa's version is called Dynamic Passcode Authentication (DPA).

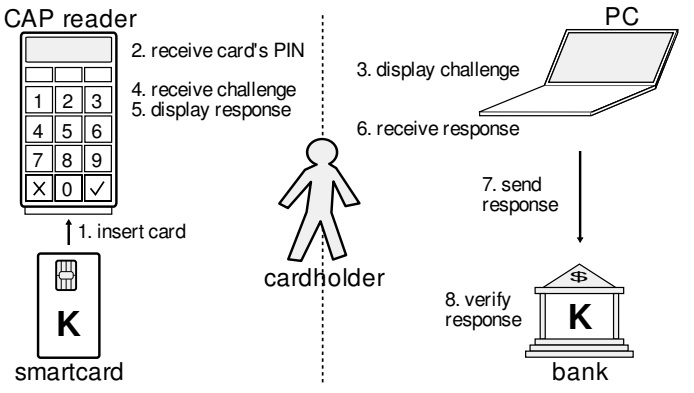

Fig. 3. In *respond* mode, after initiating an online banking session, the user inserts the card into the reader (1), keys in the PIN (2), and then enters the challenge given by the web page (3,4). When the CAP reader's response is displayed (5), the user enters it into the appropriate field on the web page (6). Since the card and bank share a cryptographic key, the bank can verify that the response is correct given what it knows about the state of the card's transaction counter (7,8).

protocol. However, based on our analysis, we have been able to generate CAP response codes and use them successfully on real bank websites. An example protocol run, collected by our protocol analyser, can be found in Appendix A.

CAP operates in three modes – *identify*, *respond*, and *sign*. These differ in the information a user is asked to enter before a response code is generated. For all three modes a PIN is required first. Thereafter, *identify* just returns a one-time code; for *respond* a numerical challenge is required; and for *sign* an account number and a value are needed. The numerical response code is a compressed version of a MAC computed by the card under its key; it is calculated over the information entered by the customer, a transaction counter, and a flag showing whether the PIN matches the one stored on the card. A *respond* transaction is shown in Figure 3.

The implementation of the CAP system is heavily based on the EMV smart card protocol being introduced throughout Europe for credit and debit card point-of-sale transactions. In the UK, EMV is known under the "Chip & PIN" brand. Using EMV as the basis for CAP reduced development and deployment costs; using the existing debit card base meant that the CAP devices themselves did not need to be personalised.

An overview of the CAP protocol flow is given below, with emphasis on how it deviates from EMV. For more information we refer the reader to the EMV specification [5].

Select application. EMV cards may be multi-application, so the reader must select the right one. The reader tests if the card is CAP enabled by searching a list of application identifiers stored on the card, and selects the first one available.

As the application identifiers for CAP are distinct from those for EMV, if a card is not CAP enabled the reader will fail to select an application and reject the card. Hence, a new generation of cards had to be issued by the participating banks before they sent CAP readers to their customers.

The application identifiers attempted by the CAP readers we have examined, in the order in which they are tried, are `0xA0000000048002`, `0xA0000000038002`, and `0xA0000002040000`. NatWest cards implement the first application, and Barclays the second. Although HBOS has not deployed CAP readers, their cards are CAP enabled and implement the second application.

Read records. Following application selection, the reader requests a list of all the data records stored by a card. These form a hierarchy, with each node being prefixed by a one or two byte *tag*. In a standard EMV transaction, these would include account number, public key certificates, signatures, and so on. With CAP, only three entries are of interest – the card data object lists (CDOL1 and CDOL2), identified by tags `0x8C` and `0x8D` respectively, and the CAP bit filter[2], identified by the tag `0x9F56`. Tag `0x9F55` is also present on cards, with value `0xA0`, but its purpose is unclear.

PIN verification. Once the reader has successfully read all available records, it prompts the customer for a 4-digit PIN. This is sent to the card as the payload to the EMV standard `VERIFY` command. If three consecutive PIN verifications fail, the card will lock itself until taken to an ATM and reset with the correct PIN. While the EMV standard allows for a transaction to continue if the PIN verification fails or is omitted, the CAP reader requires that the card accept the PIN before continuing. Surprisingly, this is a serious bug; we'll discuss the reason in Section 4.1.

Cryptogram generation. Next, the reader requests an application cryptogram from the card, using the `GENERATE AC` command. The reader first requests an Authorization Request Cryptogram (ARQC), indicating that it wishes to perform an online EMV transaction. The card then responds with an ARQC, indicating that the card is willing to do so. If this was an EMV transaction, the reader would send the ARQC to the bank for verification, but it cannot do so because it is offline. So the reader then requests an Application Authentication Cryptogram (AAC), indicating that it wishes to cancel the transaction.

A similar transaction flow might be seen during a point-of-sale transaction if a shop is only willing to accept online transactions but fails to connect to the bank (e.g. if the phone line is engaged). This protocol may have been designed so that CAP maintains maximum compatibility with EMV smart card applications. While EMV supports offline transactions by requesting a Transaction Certificate (TC) instead of an ARQC, some card risk-management algorithms may lock up if there are too many consecutive offline attempts. Cancelling the transaction should reset the smart card's risk-management parameters.

[2] The CDOL name and tag are defined in the EMV specification, but the CAP bit filter is not. We therefore had to coin our own term for it.

Table 1. Relevant CAP fields and their values

Field	Tag (hex)	Value (hex)
Terminal Country Code	9F1A	0000
Terminal Verification Results	95	8000000000
Transaction Currency Code	5F2A	0000
Transaction Date	9A	010101 for app. 0xA0000000038002, 000000 for app. 0xA0000000048002
Authorisation Response Code	8A	5A33
Other Amount	9F03	000000000000
Transaction Type	9C	00

The first and second `GENERATE AC` call is controlled by the CDOL1 and CDOL2 respectively. Each CDOL lists a series of tags, specifying which data items must be included in the command payload. The two fields used for CAP are Authorized Amount (AA – `0x9F02`), and Unpredictable Number (UN – `0x9F37`). Normally, the former would store the value of the transaction, and the latter would be a terminal supplied nonce. For CAP *identify*, both are zero; for *respond*, AA is zero and UN is the challenge; and for *sign*, AA is the transaction value and UN is the destination account number.

Other tags in the CDOL have hard-coded default values provided by the CAP reader. The values we have observed being produced by the NatWest and Barclays CAP readers are shown in Table 1.

Reader response formatting. The response to a `GENERATE AC` call includes a 16-bit application transaction counter (ATC), a Cryptogram Identification Data (CID) type code, Issuer Application Data (IAD) which includes the result of the PIN verification, and an Application Cryptogram (AC) which is a MAC over all this data. The MAC method used to calculate the cryptogram, and the structure of the IAD, are not specified by the EMV standard, as they are proprietary to the card issuer. In practice, a common choice is 3DES CBC MAC, under a session key. This session key is derived from a card master key shared between the issuing bank and the card, and the ATC. One example session key derivation algorithm, designed to resist power analysis, is described in the optional part (Common Core Definitions) of the EMV specifications [5, Book 2, Annex 1.3].

The data from the first `GENERATE AC` call and the CAP bit filter (from the read records stage) are used to generate the response code. Going through each bit of the bit filter from left to right, if the bit is a '1' the corresponding bit from the `GENERATE AC` response is kept; otherwise it is discarded. The result is a number with the same number of bits as the bit filter has '1's. Finally, this number is converted from binary to decimal with leading zeros are removed; the result is then displayed on the reader's screen. An example of this process is shown in Table 2.

Table 2. An example of a NatWest card CAP computation. The fields of the `GENERATE AC` response are the Cryptogram Identification Data (CID, 1 byte), Application Transaction Counter (ATC, 2 bytes), Application Cryptogram (AC, 8 bytes), and the Issuer Application Data (IAD, variable length).

	CID	ATC	AC	IAD
Card output	80	A52D	AD452EF6BA769E4A	06770A03A48000
Bitmask	00	001F	00000000000FFFFF	00000000008000
Filter	..	..0D	..........69E4A	.........8...
Filter (binary)	0	1|101 0|110	1|001 1|110	0|100 1|010 1
Filter (hex)			1AD3C95	
Decimal response			28130453	

We have seen the following bit filters:

NatWest	00 001F	00000000000FFFFF	00000000008000
Barclays	80 00FF	000000000001FFFF	00000000000000
HBOS	80 007F	E000000000003FFF	00000000008000

The NatWest Bank uses a bit filter that selects the five least significant bits of the ATC, the 20 least significant bits of the MAC, and one bit from the proprietary issuer application data field. Barclays uses a bit filter that selects the top bit of the cryptogram type, the least significant eight bits of the ATC, and the least significant 17 bits of the MAC. HBOS uses a bit filter that selects the top bit of the cryptogram type, the least significant seven bits of the ATC, 17 non-contiguous MAC bits, and one bit from the proprietary issuer application data field. The CID type field should always be `0x80` for an ARQC – perhaps it is selected because an AAC may be generated, or simply to ensure a leading-one and so a fixed-length response.

Response verification. Since the bank knows the input to the `GENERATE AC` call, and can reconstruct the ATC provided it knows the most significant bits not included in the response, it can repeat the MAC and check if the response entered by the customer matches the expected value.

3 Use in Online Banking

CAP provides an authentication token, but does not specify how it should be used. Each bank has made its own decision on which of the three modes to use, and the semantics of the data fields. This is problematic from a usability perspective, since the inconsistent user experience will make it easier for phishing attacks to manipulate user behaviour.

NatWest only uses the *respond* mode, with an 8-digit challenge. For money transfers, the first four digits of the challenge are random and the last four are

the last four digits of the destination account number. Where there is no account number, such as transactions to change personal details, the last four digits are '4444'. Logging into online banking does not require the CAP code, and the value of a transaction is not authenticated.

In contrast, Barclays does require an *identify* response for login. For performing a transaction, a *sign* response is required, with destination account number and transaction value entered. A significant weakness is that there is no bank-provided freshness in the transaction. While the ATC does ensure that a response cannot be replayed, the bank has no assurance that the CAP's response was generated recently.

4 Vulnerabilities

4.1 Card Theft

A serious problem is that CAP readers may be used during mugging. Since the roll-out of Chip & PIN, a criminal who has stolen a card needs to know its PIN to use it in card-present transactions. In July 2008 two French students were tortured to death in their London residence six days after it was broken into and a computer stolen. Days after the murders the police revealed that the attackers were after the students' card PINs [6]. In February 2007, two Manchester men murdered a 62 year old security guard after he refused to reveal his card's PIN [7].

Previously, muggers marched a victim to an ATM to ensure he gave them the right PIN. Now, with CAP, criminals have a portable device that will tell them if their victim is lying. While the EMV protocol always permitted such a device to be built, that requires technical skill, and wasn't in practice done. CAP has made the capability ubiquitous. It reduces the risk to muggers, as now they can keep their victims in a quiet place, and not risk being caught or seen by CCTV by going near an ATM. It would have been easy enough for the banks to design CAP without revealing the result of the PIN verification, but they failed to foresee the risk.

In our view, this was negligent: authentication tokens designed by other firms, such as the Racal Watchword (also known as Sytek PFX Passport [8]), would generate an erroneous response if the wrong PIN was entered but would not indicate this to the user, and so are not vulnerable. Worse, the two banks that have flooded the UK with CAP readers have thereby placed not only their own customers in harm's way, but have also endangered the customers of other banks who have enabled their debit cards for CAP. It remains to be seen whether customers will be able to demand cards that are not CAP-enabled and thus do not put them at needless physical risk.

There are other issues related to card theft. For example, if a customer is issued with an ATM card, the same card and PIN will be used for CAP, and so the PIN digits on the reader will wear down. Because customers are encouraged to carry their CAP readers around with them, it may be stolen along with their cards, perhaps telling the thief which digits to try. If the PIN has 4 distinct digits this leaves 24 different orderings, this increases the chance of an attacker

guessing the correct PIN in three attempts from 1 in 3333 to 1 in 8. If a customer has multiple cards with the same PIN, the attacker has even better odds.

4.2 Software Implementation

CAP was intended to offer a trustworthy user device to defeat the malware that infests ever more PCs. However, it is inconvenient for users, and prevents integration between home/office banking software and online accounts. Therefore, there is demand for a software implementation of CAP, which sends commands to a smart card connected to a PC. With some reverse-engineering effort, and access to the public EMV specifications, it is straightforward to implement this system, because the CAP readers contain no secret. We may therefore expect this demand to be met by software vendors, leading to malware-infected PCs having unfettered access to smart cards and PINs, not only opening up online banking fraud, but also allowing cloned ATM magnetic strip cards to be made and relay attacks [9] to be implemented[3].

4.3 Middleperson Attacks

A fundamental problem with smart card payment at the point of sale is that the customer has no trustworthy display to show what transaction the card is authorising. Drimer and Murdoch [9] demonstrated how this weakness could be exploited by a criminal who sets up a tampered Chip & PIN terminal, which displays one transaction, but actually is relaying the smart card communications to a counterfeit card being used for a much higher value transaction. Also, since the same card and PIN are used for ATM withdrawals, a criminal could also withdraw cash. Since CAP introduces yet another role for the smart card, a criminal with a tampered Chip & PIN terminal could generate CAP responses as well.

In current online banking, both static identifiers (i.e. username and password) and a CAP response are typically required. The risk of wide-scale attacks is limited so long as these static identifiers are not stored on the card. However, targeted attacks against high net-worth individuals – whale phishing, or whaling – are becoming a problem. One example is an attack against the Novalis Ubuntu Institute in South Africa [10]. Here, a phishing or malware attack collected the CFO's account credentials. In themselves, these are not sufficient to place a transaction, because an authorisation code is also sent to the registered account holder's mobile phone. So one criminal went to the mobile phone shop impersonating her driver, offered a counterfeit ID and the phone number of a female accomplice who impersonated the CFO herself, and requested a new SIM for the CFO's account. He used this, along with the account credentials, to empty the institute's account of R90 460 (approximately £6 000). We understand that the bank and phone company are disputing liability for this fraud.

[3] We are aware of at least one C implementation of CAP, although it has the Barclays bit filter hard-coded – `http://aa.gg/free/barclays-pinsentry.c`

A similar attack could be performed with CAP. The customer, using a tampered Chip & PIN terminal, would insert their card and enter their PIN as usual. The terminal would then generate the necessary CAP responses, and optionally also carry out the legitimate transaction. Shortly after, the customer would receive a personalised phone call or email, stating that a suspicious transaction had been noticed (stating the shop name they just used), asking for their online banking credentials. Since Barclays only uses *identify* and *sign* mode, there is no server-provided freshness or a timestamp, so the previously collected responses can be used, provided the customer had not logged into online banking in the meantime. With NatWest, which uses *respond*, there is a server-provided nonce, so the fraudulent transaction has to be in near real-time, and account credentials would need to be collected before the CAP responses were generated. The banks could resist this problem by offering separate CAP-only cards, but NatWest refused to do so for one of us.

CAP has also been proposed for authenticating online purchases, through the "Verified by Visa" and "MasterCard SecureCode" schemes. Here the problem might be even worse, as most if not all the details needed for an online purchase are stored or printed on the card. CAP is also being rolled out for authenticating citizens to the "Government Gateway", a single sign-on system for accessing UK government services [11]. Currently the government are believed to be issuing cards specifically for this purpose, so the relay attack above would be resisted, but if they try to optimise by sharing the existing card base then attacks could be expected.

4.4 Supply-Chain Infiltration

CAP was unpopular when introduced [12], as customers did not want to have a reader for each bank, or have to carry them around to use online banking both at home, work, or while travelling. Customers were reassured that other banks' readers are compatible, and they can use another person's reader if they do not have their own. This behaviour makes it easy to infiltrate the CAP supply chain. For example, CAP readers are available for sale on eBay – a criminal could tamper with them so that they copy the chip details (which on many cards includes a copy of the magnetic strip) and record the PIN.

Later, the CAP reader could disable itself, so the owner will send it back to the seller for a refund. The criminal could then make fraudulent ATM withdrawals abroad where magnetic strip transactions are accepted. The CAP reader could also prompt for other details, such as the printed CVV code or online banking credentials, for use in the attack described in Section 4.3. An enhancement to the attack would be to send a CD with the CAP reader, which auto-installs malware to collect online banking credentials. Criminals could even install a compact GSM module into the CAP for sending back information in real-time. The police have already found Chip & PIN terminals that have been tampered with during or soon after manufacture and contain GSM mobile phones to send card and PIN data to criminals [13].

4.5 Social Engineering

The security of CAP depends on users properly understanding the semantics of the data they are being asked to enter into the reader. That is, for Barclays, the customer must verify the destination account number from a trustworthy source before entering it. For NatWest, the customer must verify that the last four digits of the challenge provided to them by the website match the last four digits of their desired destination bank account number. These instructions are complicated, unintuitive, and not made clear to customers. It is therefore likely that a phishing website could induce a customer to enter fraudulent details into the CAP reader, and send the response to the attacker. This exploit is made even more likely by the vague prompts for each data item. Rather than asking for the destination account number for a payment, the Barclays CAP reader simply displays 'REF:'.

4.6 Protocol Weaknesses

The CAP protocol has been highly optimised to reduce the amount of information customers need to enter and to maximise backwards compatibility – this has introduced vulnerabilities. For example, the lack of server freshness allows CAP responses to be requested long before they are needed, as described in Section 4.3. Another flaw is the overloading of the Unpredictable Number field of the input to `GENERATE AC` command: in *respond* mode it is the challenge, but in *sign* mode it is the destination account number. This means that a CAP response in *sign* mode for a zero transaction is a valid *respond* mode response.

An attacker could use this property in a social engineering attack, to defeat customers who are trained to be suspicious about *respond* mode. By asking the victim to perform a 'test transaction' to a dummy account, and assuring them the value is £0 so they are safe, the attacker can get a valid response and use it for fraudulent purposes. Currently, the risk of this attack is low, because only the Barclays CAP reader accepts a £0 transaction, and Barclays do not currently use *respond* mode – it is unclear whether this is by design or fortuitous accident. However it does illustrate the fragility of the protocol, and the failure to follow accepted design principles such as type explicitness [1].

Another example of excessive optimisation is in the NatWest protocol variant, of including a nonce as the first four digits of the *respond* mode challenge. Initially, the server provided nonce appears to defeat the attack in Section 4.3, because the nonce cannot be discovered without getting the online banking credentials first. However, there is a time-space tradeoff – if the shop's malicious Chip & PIN terminal requests a large number of responses from the card (with random nonces), and then later requests a sufficient number of challenges from the online banking site, there will be a nonce collision and so he would know a valid response. With 100 challenges and responses, the probability of success is approximately 63%.

The bank website could check for excessive transaction counter gaps, or limit the number of challenges generated. However, the card's transaction counter is

incremented merely by inserting it into a reader, so fairly large gaps will be common and locking accounts on this basis would increase support call costs. In fact, after deploying CAP, banks have removed other protections – Barclays lifted their transaction limit from £1 000 to £10 000. Even if this attack is currently detected, the small nonce creates a fragile protocol and a minor website update may open the vulnerability again.

5 Fixing the Vulnerabilities

The basic principle behind CAP – a trusted user interface and secure cryptographic microprocessor – is sound. However the system has been optimised literally to death. Re-using ATM cards for point of sale and CAP saved money but created a vulnerability to relay attack, and increased the risk of violent mugging and murder. Omitting a server-provided nonce removed assurance that responses are freshly generated. Overloading fields introduce a social engineering vulnerability, as it makes the system model too complex for the average user to be expected to visualize.

The type confusion between *respond* and *sign* could be fixed on the CAP reader by including a response-type flag in the `GENERATE AC` input. Also, the time-memory attack against nonce guessing could be mitigated by a narrower window for acceptable values of the ATC. However, the other flaws require a more substantial re-design. The mugging vulnerability is a side-effect of the EMV design – a PIN can be checked by the card itself, with no authentication. Adopting the Racal Watchword approach, of returning an erroneous response if the wrong PIN is entered, would fix this problem, but harm usability.

The German CAP variant, TAN generator (HHD 1.3) [14], incorporates defences against a number of the attacks we discuss in the paper. The challenge displayed by the bank website includes a prefix which customises the user prompts. This reduces the risk of social engineering because the field descriptions are more specific (e.g. 'account number' or 'IBAN', rather than 'REF'). The prefix is also incorporated in the response calculation, fixing the type confusion vulnerability. All types of challenges may include a random nonce (up to 7 digits), providing an assurance of freshness. Finally, PIN verification by the card is optional, reducing the risk from mugging.

One error in CAP appears to have been trying too hard to reduce the number of characters the user has to type. This is the root cause of several vulnerabilities. Only including the last four digits of the account number in the NatWest system increases the risk of a fraudster having a matching or similar account. This, and the inadequate or missing nonce, could be resolved by having a higher bandwidth channel between the computer and CAP reader, so not requiring the customer to re-type the transaction, allow full account details to be displayed, and permit a large nonce to be incorporated in the response.

One example of a high-bandwidth channel is the USB-connected FinTS (Financial Transaction Services) class 3 smart card reader, incorporating a keypad and display [15]. This would be problematic for use in Internet cafés, inconvenient to carry, and may require complicated driver installation. The Cronto

transaction authentication system [16] uses the visual channel, generating a specialised barcode, read by a camera phone or a dedicated client device. As with a class 3 smart card reader, full transaction details are displayed without the inconvenience and security implications of manual input, but it requires no physical connection to the PC. A PIN may optionally be used, and as with Racal Watchword, it does not provide confirmation to a mugger if the entered PIN is incorrect. In addition, customers could be given a duress PIN [17] (as offered with RSA SecurID) which permits access to the system but that triggers an alarm at the bank.

Making it harder to implement CAP in software would also have been desirable for security. Making the specification secret was insufficient as it could be reverse engineered, so following Kerckhoffs' principles [18], a key should be embedded in the CAP reader, which is used to HMAC the response. If the key is global across all readers, there is a risk of compromise, even if stored in tamper-resistant memory, so a key revocation procedure would be needed. Switching to a per-device key would be more secure. It would prevent customers from sharing CAP readers, between banks or customers, or buying them off eBay but this may in fact be beneficial, as discussed in Section 4.4.

6 Policy Implications

In many respects, CAP is an improvement over the existing static password scheme. However, it may not be beneficial to customers because while banks are liable for fraud due to forged signatures, there is no statutory protection for the victims of electronic fraud [19]. UK banks have also recently changed the voluntary code of practice – the Banking Code – to make customers liable for fraud if they do not have up-to-date anti-virus and firewall software [20]. Having deployed a new security system, even with weaknesses, the banks have further reduced customer protection.

While the Banking Code does state that the bank must show that the customer is liable, it does not say what evidence the bank must record, what evidence is sufficient to prove liability, and who the proof must be presented to. In practice, where the case is heard by the Financial Ombudsman Service, the bank merely has to claim that a chip was read and a PIN was used [21], and the evidence used to reach this conclusion will be kept secret from the customer. We may expect a similar position to be taken when PINs are used for online banking too.

This shift in liability is particularly problematic because the specification of CAP is secret, and it is not subject to any public certification procedure. In contrast, the EU Digital Signature Directive requires Common Criteria certification, which implies a public certification report. It also requires the full transaction be authenticated, through a dedicated trusted display. This would however have cost maybe $100 per device. Instead, the banks have optimised the design, and this reminded us of the late Roger Needham's description of optimisation which we quoted in the introduction.

Recent events in financial markets have highlighted shortcomings in banking regulation in Europe and elsewhere. Here then is another shortcoming: the regulators should not have believed the banks' security models any more than their models of asset pricing and risk. In particular, regulators should not have simultaneously allowed banks to transfer liability to their customers and optimise the security engineering.

Acknowledgements

We thank the anonymous reviewers, Nicholas Bohm, Mike Bond, Joseph Bonneau, Richard Clayton, and Boris Hemkemeier for valuable comments and suggestions. Markus Kuhn and Xilinx have contributed hardware for our experiments. Saar Drimer's research is funded by Xilinx, Inc. Steven Murdoch is funded by the Tor Project and employed part-time by Cronto Ltd.

References

1. Anderson, R.J., Needham, R.M.: Robustness principles for public key protocols. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 236–247. Springer, Heidelberg (1995)
2. APACS: Online banking usage amongst over 55s up fourfold in five years (August 2007), http://www.apacs.org.uk/media_centre/press/08_24_07.html
3. APACS: APACS announces latest fraud figures (September 2008), http://www.apacs.org.uk/APACSannounceslatestfraudfigures.htm
4. RedTeam: iTAN online-banking security system. CAN-2005-2779 (August 2005), http://www.redteam-pentesting.de/advisories/rt-sa-2005-014.txt
5. EMVCo, LLC: EMV 4.1. (June 2004), http://www.emvco.com/
6. Taylor, M.: Police think French pair tortured for pin details. The Guardian (July 2008), http://www.guardian.co.uk/uk/2008/jul/05/knifecrime.ukcrime
7. Jenkins, R.: 'brainless thugs' get life term. The Times (May 2008), http://www.timesonline.co.uk/tol/news/uk/crime/article3850647.ece
8. Wong, R.M., Berson, T.A., Feiertag, R.J.: Polonium: an identity authentication system. In: IEEE Symposium on Security and Privacy, p. 101 (1985)
9. Drimer, S., Murdoch, S.J.: Keep your enemies close: Distance bounding against smartcard relay attacks. In: USENIX Security Symposium (August 2007)
10. Finn, C.: MTN not budging on fraud issue. IOL technology (May 2008), http://www.ioltechnology.co.za/article_page.php?iSectionId=2885&iArticl%eId=4402087
11. Lomas, N.: Government gateway 2.0 looks to fatter future. silicon.com (October 2007), http://www.silicon.com/publicsector/0,3800010403,39168629,00.htm
12. Make Card Readers Optional (2008), http://www.stopthecardreaders.org/
13. Samuel, H.: Chip and pin scam 'has netted millions from British shoppers'. Telegraph (October 2008), http://www.telegraph.co.uk/news/newstopics/politics/lawandorder/3173346%/Chip-and-pin-scam-has-netted-millions-from-British-shoppers.html

14. ZKA: ZKA-TAN-Generator: Belegungsrichtlinien für die Dynamisierung der TAN (January 2008), http://www.hbcizka.de/dokumente/spezifikation_deutsch/Belegungsrichtli%nien%20TANGenerator%20ve1.3%20final%20version.pdf
15. Rütten, C., Bachfeld, D.: Ausweispflicht: Sicheres home-banking mit der chipkarte. c't (17), 98–103 (2008), http://www.heise.de/kiosk/archiv/ct/08/17/098_Ausweispflicht
16. Cronto: Products datasheet, http://www.cronto.com/download/Cronto_Products_Datasheet.pdf
17. Davida, G., Frankel, Y., Tsiounis, Y., Yung, M.: Anonymity control in E-cash systems. In: Luby, M., Rolim, J.D.P., Serna, M. (eds.) FC 1997. LNCS, vol. 1318, pp. 1–16. Springer, Heidelberg (1997)
18. Kerckhoffs, A.: La cryptographie militaire. Journal des sciences militaires 9, 5–38 (1883)
19. Bohm, N., Brown, I., Gladman, B.: Electronic commerce: Who carries the risk of fraud? The Journal of Information, Law and Technology (3) (October 2000), http://www2.warwick.ac.uk/fac/soc/law/elj/jilt/2000_3/bohm/
20. Banking Code Standards Board: The banking code (March 2008), http://www.bankingcode.org.uk/
21. Drimer, S., Murdoch, S.J., Anderson, R.: Thinking inside the box: system-level failures of tamper proofing. In: IEEE Symposium on Security and Privacy, Oakland, May 2008, pp. 281–295 (2008)

A Annotated Protocol Log

Collected from a NatWest reader and card performing a *respond* computation (ISO 7816, T=0 protocol). Personal details have been redacted.

```
Command: 00a4040007 (select application)
Proc:    a4
Data:    a0000000048002
Proc:    61
Status:  6112 (more data available)

Command: 00c0000012 (application selected)
Proc:    c0
Data:    6f108407a0000000048002a5055f2d02656e
Proc:    90
Status:  9000 (OK)

Command: 80a8000002 (initiate transaction)
Proc:    a8
Data:    8300
Proc:    61
Status:  6108 (more data available)

Command: 00c0000008 (transaction initiated)
```

```
Proc:    c0
Data:    8006100008010100
Proc:    90
Status:  9000 (OK)

Command: 00b2010c00 (get static data length)
Proc:    6c
Status:  6c57 (wrong length)

Command: 00b2010c57 (read static data)
Proc:    b2
Data:    7055
          8e0a 00000000000000000100 (CVM list)
      9f5501 a0 (unknown)
      9f5612 00001f00000000000fffff00000000008000 (bit filter)
      8c15 9f02069f03069f1a0295055f2a029a039c019f3704 (CDOL1)
      8d17 8a029f02069f03069f1a0295055f2a029a039c019f3704 (CDOL2)
Proc:    90
Status:  9000 (OK)

Command: 80ca9f1700 (get PIN try counter length)
Proc:    6c
Status:  6c04 (wrong length)

Command: 80ca9f1704 (get PIN try counter)
Proc:    ca
Data:    9f170103 (3 remaining tries)
Proc:    90
Status:  9000 (OK)
```

PIN entered

```
Command: 0020008008 (verify PIN)
Proc:    20
Data:    24xxxxffffffffff
Proc:    90
Status:  9000 (OK)
```

Challenge entered: 12345678

```
Command: 80ae80001d (generate AC)
Proc:    ae
Data:    000000000000000000000000000080000000000000000000000012345678
Proc:    61
Status:  6114 (more data available)
```

```
Command: 00c0000014 (return ARQC)
Proc:    c0
Data:    8012800042b7f9a572da74caff06770a03a48000
Proc:    90
Status:  9000 (OK)

Command: 80ae00001f (generate AC)
Proc:    ae
Data:    5a33000000000000000000000000000080000000000000000000012345678
Proc:    61
Status:  6114 (more data available)

Command: 00c0000014 (return AAC)
Proc:    c0
Data:    80120000424f1c597723c97d7806770a03258000
Proc:    90
Status:  9000 (OK)

Response returned: 4822527
```