A New Approach to E-Banking

Matthew Johnson and Simon Moore
matthew.johnson@cl.cam.ac.uk, simon.moore@cl.cam.ac.uk
University of Cambridge

Abstract

This paper demonstrates that general purpose computing devices, including mobile phones, are not a sufficiently trustworthy platform for financial transactions. Current defences against phishing attacks, including multi-factor authentication systems, do not work against many attacks. Such attacks have been seen in the wild as well as in theory. This paper proposes hardware which would secure internet transactions without requiring any trust to be placed in a general-purpose computing device. The key to the device is a set of protocols to provide end-to-end security coupled with a trustworthy user interface which provides transparency about what transaction the user is authorizing. In addition, this paper proposes that the device can provide transaction audit trails which support the customer in case of disputes with the bank.

1 Introduction

The vast majority of phishing attacks seen today are very simple [42, 16]. They simply harvest static authentication details for banking websites for later use by the criminals which run them. This has led to much of the deployed protection techniques only being designed to prevent those attacks. However, that these are the most common attacks does not imply that they are the only attacks which the criminals can perform. In fact, the prevalence of simple phishing attacks is entirely down to the fact that they work. Since the rise of these naïve defences, several groups have already adapted to counter them [19, 30, 20, 21] and continue to attack the banking system.

Several more complicated schemes [36, 14, 8, 18, 43, 9] address slightly more attacks, but none of them protect users against a full middle-person (MITM) attack using a compromised terminal. In this context, a middle-person attack involves redirecting the user's traffic to their e-banking website via the attacker's site. When the user tries to perform a legitimate transaction, details such as the destination account number and value of the transaction are rewritten before forwarding them to the bank. Responses which show the rewritten destination are also changed before the user sees them and nothing appears amiss.

1.1 Leveraging Devices

Since personal computers have proven not to be a secure basis for transactions several companies and researchers have proposed systems which leverage other devices which could be considered more secure. These are commonly multi-factor authentication schemes which use not only a password to authenticate, but some other form of authentication, usually a token.

1.1.1 Tokens

The family of RSA SecurID products [39] and transaction authorization numbers [3, pp.13] are variations on one-time-passwords. SecurID is time-based and TANs are static. Static one-time-passwords are susceptible to harvesting of the codes [30]. RSA provide several forms of OTP tokens. The most basic displays a time-dependent code which the user types into a web page. More advanced tokens have PIN input to the device or a USB interface to automate the code input. All of them fail to protect against MITM attacks [22] and the USB interface allows a malicious program to request codes from the device without the user's knowledge.

One form of multi-factor authentication which banks are starting to introduce derives from the EMV (Chip and PIN) specification and is called Chip Authentication Protocol [26]. This is currently being rolled out in the UK by Barclays [15]. CAP uses the smart card in credit and debit cards to answer a challenge from the server using a hand-held device like a calculator and using the chip in the card to create a MAC using the secret key it shares with the bank. This approach is not without flaws [5]. There are two modes currently suggested for use with CAP. The first one just uses it to generate a random challenge, which is clearly not resistant to online MITM attacks. The second mode is better, asking the user to input some or all of the destination account number. This is primarily used when setting up mandates and so leaves open attacks on other transaction types.

1.1.2 Mobile Phones

Another approach which has been used is to use the SMS service as a secure side channel [17]. In this system a challenge is sent over SMS which the user must enter into their computer.

Mobile phones look like a good platform for a secure side channel. They are small, portable and ubiquitous. However, the continuing convergence of functions onto mobile devices has resulted in mobile phones being essentially general purpose computing devices and suffering from the same vulnerabilities as personal computers. Several viruses, worms and Trojans have been produced for Symbian [24, 27] and WinCE [23] (the two most popular smart phone operating systems) as well as an exploit for the iPhone [28]. While some of these are just proofs of concept, history has shown that such things quickly turn into real outbreaks, particularly when there is money to be made. If mobile phones become key in financially sensitive transactions they will quickly become a target for virus writers and those who pay them. As mobile phones continue to accrue functionality, the complexity (and therefore scope for security-sensitive bugs) increases massively.

All this suggests the conclusion that mobile phones are not a long term solution for securing financial transactions. In addition, sites like FakeMyText.com[1] also do not encourage trust in the SMS network. One bank will even send challenges via email [7]. Finally, the recent scandal of wiretapped Greek mobile phones [34], while a very sophisticated attack which probably used an insider, proves that attacks on mobile phone networks are possible.

Pietro, Gianluigi and Strangio [33] and Parno, Kuo and Perrig [31] suggest the use of portable computing devices (such as mobile phones) which are in common use and have short-range wireless built in as a security token in for two-factor authentication. The former system requires the user's PC to authenticate their mobile device before proceeding, but the bank's system is only tangentially involved in this (in that it generates the secrets for the mutual authentication). It is resistant against a number of common attacks, but it is not clear that active MITM is protected against, nor is compromise of either the mobile device or the terminal.

In the latter case they offload part of the TLS verification to the mobile computing device.

[1] http://www.fakemytext.com/

This approach successfully stops the standard MITM approach and at first glance is better than simply using TLS client certificates in the browser combined with better certificate checking. However, since the TLS session once setup is entirely accessible by the client; a compromised machine could MITM the content of the session, replacing transaction requests with others and masking the replies. It also seems possible to create a second TLS connection in the background connecting to the bank and injecting arbitrary content while the browser actually connects to the attacker.

1.2 Contributions in this paper

This paper proposes that using a secure device in order to enhance the security of general-purpose computing devices can solve many of the vulnerabilities in online transactions. The evidence above suggests that mobile phones do not constitute such a device and it is clear that much more protection can be afforded than the current schemes provide. Therefore, the contribution presented here is to use a dedicated device with a trusted user interface as a counter-measure to the attacks possible in current online banking solutions.

2 The Secure Banking Dongle

The device which this paper proposes has been dubbed a banking 'dongle'. The form-factor which is envisaged is of a small, USB-attached device with a display and one or two buttons. USB is now the standard interconnect for computer peripherals and is supported in all major operating systems. All of the security for the system would reside in the device, which would have a modicum of tamper-resistance.

Alternative form-factors are possible. Bluetooth is the current short-range wireless protocol of choice and would provide a suitable connection to a host PC. An alternative which is popular in Japan is the 2D bar code [35] which can be read by a camera on a mobile phone. This has the advantage of not needing any special hardware support on the client (USB is almost ubiquitous, but is disabled in some workplaces for security policy reasons). A version of this is being developed by Cronto[2] for exactly this use in their security products.

2.1 Device IO

The device presents a trusted user interface to the customer. This requires a screen to display details of the transactions and some method of authorizing or denying transactions.

The minimal implementation of this is a pair of 'OK' and 'Cancel' buttons. Since performing transactions also requires logging into the bank's website with traditional credentials this is already a two-factor authentication scheme; but this can be improved upon with the trade-off being increasing the cost of the device.

One option would be to include a PIN-pad and require entering a PIN to confirm a transaction. Another option would be to use biometrics, probably in the form of a fingerprint scanner in the device. Both of these would reduce the exposure from a stolen device but are orthogonal to the rest of the ideas presented here.

2.2 Low-cost Device

One of the key premises of this research is that the result could be a consumer device. This requires it to be producible cheaply and in bulk.

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

The canonical example of a cheap device held by consumers is the smart card. These are sufficiently cheap that banks can issue them to all customers. Producing something this cheap would be extremely difficult; however, this may not be necessary.

There are two approaches for increasing the price-point at which such a device is viable. Firstly is billing it as a 'platinum' option for their high-value customers. Those customers are often more aware of the issues and would value a bank which offered such a device.

Secondly it could be offered as an after-market third-party offering which the consumer purchases. If enough benefit can be provided for the consumer it would be a viable proposition. See section 3 for some ideas on how to make it attractive for the consumer.

Such a USB-attached device with a display is comparable in complexity, and therefore price, to USB internet phones which typically retail at around $20–$50 [32, 40]. Alternatively, Crystalfontz[3] sell a number of USB-attached LCD displays with keypads for $30–$60.

Oikonomakos, Fournier and Moore have developed cryptographic hardware in polysilicon [29]. While currently just proof-of-concept, the ability to produce significant logic along with TFT driver circuitry should provide a simple solution with few components for the device which has been proposed. This will be sufficiently cheap to produce in bulk that it will be attractive to both banks and consumers.

2.3 Transaction Transparency

In 2001 Anderson [4, pp.24–25] wrote about the chosen protocol attack. This has particular relevance because the example he used, if only in the abstract, was with online shopping. A chosen protocol attack boils down to an authorization which the user thinks is for one purpose actually being used for a different purpose. Systems like CAP show that financially-sensitive authenticators are being used in multiple systems and can therefore be vulnerable to the chosen protocol attack.

Solutions for this involve ensuring that messages are unambiguous as to their purpose and destination and not allowing third parties to use the same authentication system. The problem with this is that there is a growing desire from consumers and merchants for a single system which they can all use. In addition, the unambiguity of the message must be clear to the user, not just to the computers involved. This poses a problem when the assumption is that the computer in use is compromised.

This lack of transparency is at the root of all phishing and pharming attacks. If the user could be sure they knew what they were authorizing these frauds would not be possible.

2.4 Connecting the device

In this paper it is assumed that the computer the device is connected to is compromised or, more accurately, that it is part of the untrusted 'network' between the two principals. If the protocols are designed such that no trust relies on the computer, then the method of connecting the device to the server is irrelevant. It is useful, however, to assume some properties of the connection, even if the security doesn't rely on them. In particular it is useful to assume a reliable transport layer, at least in the absence of any malicious intervention.

At the physical level the most obvious form-factor for connecting to the host PC is via USB. Since this is to be used to secure Internet banking, the host PC will already be using TLS-secured HTTP [10] to connect to the bank's website. An application needs to be provided which will forward the protocol messages from the HTTP connection to the USB-connected

[3] http://www.crystalfontz.com/

device. TLS encryption does not need to be used here and would increase the complexity of the device.

2.5 Protocols

The transport will forward the protocol messages from the device to the bank's server. The protocol for each transaction can be quite simple, although there are many issues with key setup which are not addressed in this paper.

The protocols given here assume that a good block cipher is in use (AES [1] is a suitable candidate, but the protocols are by no means limited to that) in a suitable mode (including, but not limited to, CBC [12] with a random IV). The Message Authentication Code used is also not specified but assumed to be good. CMAC [13] is a candidate for this. They also assume that there is already a shared key between the device and the bank. This is practical in the case where the device is provided by the bank to its platinum customers.

$$
\begin{aligned}
M_1 = &\quad I, \text{“}INIT\text{”}, Len, D, K_{BD1}, K_{BD2} \\
B \rightarrow D: &\quad \text{“}INIT\text{”}, Len, IV, \{M_1\}_{E_{K_{LT}}}, MAC_{K_{LTM}}(M_1) && (1) \\
M_2 = &\quad I+1, \text{“}ACK\text{”}, Len, D \\
D \rightarrow B: &\quad \text{“}ACK\text{”}, Len, IV, \{M_2\}_{E_{K_{BD_1}}}, MAC_{K_{BD_2}}(M_2) && (2) \\
M_3 = &\quad N, \text{“}TRANS\text{”}, Len, D, \text{transaction}, Type \\
B \rightarrow D: &\quad \text{“}TRANS\text{”}, Len, IV, \{M_3\}_{E_{K_{BD1}}}, MAC_{K_{BD2}}(M_3) && (3) \\
D \rightarrow U: &\quad \text{transaction} && (4) \\
U \rightarrow D: &\quad Auth && (5) \\
M_6 = &\quad N+1, \text{“}AUTH\text{”}, Len, D, \text{transaction}, Type, Auth \\
D \rightarrow B: &\quad \text{“}AUTH\text{”}, Len, IV, \{M_6\}_{E_{K_{BD1}}}, MAC_{K_{BD2}}(M_6) && (6)
\end{aligned}
$$

Figure 1: Banking Dongle Transaction Protocol.

Figure 1 shows the protocol. The notation used is fairly standard. Messages are given as source $\rightarrow$ destination and then a list of the message fields. Encryption is denoted by braces subscripted by E and the key used for the encryption. MACs are denoted by MAC subscripted by the key and then parentheses containing the data to be used to calculated the MAC.

The principals involved in the protocol are the bank (B), the device (D) and the user (U).

Message (1) in this protocol is the session-key initialization message. It is encrypted under the long term shared key (K_{LT}) and with a MAC using the long term shared MAC key (K_{LTM}). It is sent at the start of each session and contains the message type and length (Len), two block cipher keys (K_{BD1}, K_{BD2}), the name of the destination device (D) and an incrementing counter (I) identifying this run of the protocol and the keys used to prevent replay attacks. These keys are used for encryption and MAC respectively in the rest of the protocol. This message is acknowledged by message (2) which confirms the receipt of the keys by encrypting $I+1$ under the session keys.

Messages (3) and (6) perform a transaction. These messages contain the message type and length, a block encrypted under the session key, a MAC of the encrypted block under the session MAC key and the IV for both.

The transaction request message (3) contains in the encrypted block a copy of the plaintext data plus the destination ID to prevent splicing attacks, the length of the transaction data,

data describing the transaction taking place and the type of transaction. There is also a nonce (N) to provide freshness guarantees, which is unique to this run of the protocol between the device and the bank. The description of the transaction will include any information necessary to describe the transaction. Typically this will be the amount, identifiers for the source and destination and the unit of currency.

Between the last two messages in the protocol between the device and the bank the user is shown the transaction details on the trusted UI. They then have to make an authorization decision on the transaction and either confirm or deny the transaction using a button on the device.

The response (6) repeats all the information from the request including the nonce incremented by one and also inputs an authorization code to indicate whether the transaction should proceed. Again there is a MAC to ensure integrity of the message.

2.6 Security Analysis

Here follows an informal security analysis of the above protocol.

2.6.1 Attacker Model

It is assumed that the attacker has complete control over the end user's computer and all communications links between the bank and the user. Thus he is able to observe and modify any messages as well as performing more high-level attacks such as DNS spoofing and the sending of fake messages.

2.6.2 Passive Attacks

The first type of attacks are passive attacks. These are ones in which the attacker merely observes the messages. They are the easiest to protect against, and hence the least useful. The goal of an attacker in this scenario is to recover the key in order to send fake messages or to infer some information about the messages.

The messages are all encrypted in cipher-block chaining mode with a random initial value and a nonce or ID in each message to ensure that all messages are different and no inference can be drawn from identical repeated messages or parts of messages. The cipher chosen should prevent any other attacks; however, to prevent too much ciphertext being generated under the same key transactions are encrypted under a session key which is regenerated each session.

2.6.3 Active Attacks

Active attacks cover every situation in which the attacker alters the message flow. This includes inserting and deleting messages and modifying existing messages.

Messages are protected by a message authentication code with a different key from the encryption which is also generated for each session. The MAC provides integrity protection against modifying the message.

Message insertion and replay attacks are the main threat against the protocol. All the messages include their message type and destination along with a unique value for that run of the protocol. This prevents messages being used in different runs of the protocol from intended.

Starting new protocol runs as an attacker with replayed messages is prevented for key initialization by including an incrementing counter for each run of the protocol. The bank keeps track of these and aborts protocol runs with repeated counters.

Duplicate transaction request messages may be sent and will be accepted by the device, however they will not correspond to an outstanding transaction when a reply is received by the bank and will have no effect. This is similarly true of repeating the transaction response messages.

Finally there is the attack of deleting messages. An attacker can simply perform a denial of service attack by dropping some or all of the messages in the protocol. In most cases it will be obvious to the user that a denial of service is happening, however, if the attacker drops the last message in the protocol and then forges a 'transaction complete' message on the PC they may believe the transaction has completed. A possible solution to this would be to add in a seventh message which displays the transaction confirmation on the device. However, it is not obvious whether such an attack would be of any gain.

The key initialization acknowledgement message ensures that deleting the session-key initialization message is noticed early in the protocol run by the bank. Stopping new session keys being used by the device cannot cause more cipher text to be sent under the same key since it will only respond to a valid message from the bank (which will not be sent). Generating an extra response to an existing message through replay with a different authorization from the user is possible, but since a random IV is chosen nothing can be inferred from the two messages. Deleting any of the initialization, acknowledgement, transaction request or response messages will merely cause the protocol run to fail.

2.6.4 Cipher Attacks

This paper assumes that the particular cipher chosen will be secure against known attacks. Since no particular cipher is given here it is not possible to comment on specific attacks. However, it is good security practice to reduce the attack surface where possible. Using CBC-mode with a random IV ensures that repeated plaintext does not result in repeated ciphertext [37], which is a desired security property.

2.7 Use of the protocols

An important part of protocols is defining how they should be used; many good protocols are broken because they are used badly in practice. Therefore, the following additional restrictions are placed on use of the protocol.

- Session Key Lifetime: Key initialisation should be performed each time the client applet connects to the bank.
- Failure Handling: All messages whose MAC fails to validate, the protected data doesn't match the public data or the destination doesn't match the receiver's ID should be dropped without acknowledgement.
- Key ID handling: Session key IDs must only ever increase.
- Replay Protection: Banks should discard all messages other than transaction responses with a valid MAC, matching public/private data and a nonce which corresponds to an outstanding transaction request.
- Transaction Overload: A transaction should either be authorized, declined or timeout before displaying another request. New transactions should not be accepted within a given time of the previous request.

The protocol is designed to be used in a system where the end station is compromised. Therefore, transport layer security between the server and the applet, such as TLS, does not help. Any attacks which could be made on an unsecured transport can also be made on the end-station after decryption. However, since the technology is already in place there is no penalty for using it and it adds an extra layer of defence against some classes of attack. Defence in depth is a good security principle since it allows for failures in parts of the system without this compromising the whole system.

With that in mind there are several other recommendations which can be made regarding the use of these protocols.

- The connection between the applet and the server and the web browser and the server should be TLS encrypted with a certificate signed by a trusted certification authority.
- The applet should be given a key in parameters from the web page which is passed to the server with requests.

These recommendations prevent most of the attacks possible without compromising the user's terminal including in particular denial of service attacks which would otherwise be possible on our protocol. This denial of service is still possible with a compromised end-station; but that is always going to be the case.

3 Protecting the Consumer

While proposing the 'Electronic Attorney' [6], Anderson and Bond make the observation that in the current banking system there are devices which protect the interests of the bank and devices which protect the interests of the merchant but that these interests do not always align with those of the consumer. The 'Electronic Attorney' would be used in combination with a credit card in order to protect the interests of the consumer.

If consumer has a device which takes part in the transaction process then it can also protect the consumer's interests.

3.1 Audit Logs

When consumers are the victims of fraud the banks are meant to refund their money. However, when the transaction has been authorized by PIN, banks claim that it is the responsibility of the customer. Drimer and Murdoch [11] have shown a number of attacks which produce PIN-authorized transactions and the banks have been unwilling to provide evidence that the PIN authorizations are valid. There are also suggestions that this might be applied to Internet transactions via the Securecode/Verified by Visa schemes.

Therefore, it would help if the consumers could be provided with an audit trail which they could show to prove they had not made the transaction. This would level the playing field between the consumers and the banks. To that end, the device should produce and audit log which will verify that the contents is both complete and accurate.

3.2 Audit Protocols

To avoid increasing the secure storage requirements on the device the audit messages once created form continuous trail which can be stored on a untrusted medium and as long as it is unaltered can be verified. This allows this to be sold as a third-party service in which the only

trust relationship necessary is that the consumer trusts the provider to keep an accurate record of the log. Once produced anyone verifying the log does not need to trust the provider.

Figure 2 shows a version of the protocol adapted to produce a log. This is similar to the protocol presented by Schneier [38], however, this protocol does not consider the bank to be trusted to look after the consumer's interest, only its own. Integrating the logging with the transaction protocol results in the bank being part of the authentication of the log. this is discussed in more detail in section 3.3.

$$
\begin{aligned}
M_6 = \quad & N+1, \text{“}AUTH\text{”}, Len, D, h(O_{I+1}), h(L_{I-1}), \\
& \text{transaction}, Type, Auth, MAC_{K_D}(\text{transaction}) \\
D \rightarrow B: \quad & \text{“}AUTH\text{”}, Len, IV, \{M_6\}_{E_{K_{BD1}}}, MAC_{K_{BD2}}(M_6) \qquad (6) \\
M_7 = \quad & MAC_{K_B}(N, I, \text{“}TACK\text{”}, Len, D, \text{transaction}, \\
& h(O_{I+1}), h(L_{I-1}), Type, Auth, MAC_{K_D}(\text{transaction})) \\
B \rightarrow D: \quad & \text{“}TACK\text{”}, Len, IV, \{M_7\}_{E_{K_{BD1}}}, MAC_{K_{BD2}}(M_7) \qquad (7) \\
L_I = \quad & N, I, Len, Type, \text{transaction}, M_7, \\
& h(L_{I-1}), h(O_{I+1}), O_I, Auth \\
D \rightarrow L: \quad & L_I, MAC_{K_D}(L_I) \qquad (8)
\end{aligned}
$$

Figure 2: Audit protocol

The same notation is used as in figure 1. The principals are the bank (B), device (D) and the audit log (L). Log entries are denoted by L_I and the nonce for that entry as O_I. Nonces denoted by O_I are used as commitments between log entries and form part of the hash chain within the log; they are completely independent to the nonce N used in the protocol run between the bank and the device. The new cryptographic primitive in use here is a hash function, denoted by $h()$. This should be a secure hash function such as SHA-256 [2].

Messages (1) through (5) are the same as those in figure 1 and have therefore been omitted for brevity. In message (6), the device includes in the transaction authorization a hash of the previous transaction and of the nonce for the next transaction. Message (7) is the bank acknowledging the transaction and in the process validating the completeness of the log.

The final message is the log entry which is stored. It contains the transaction, the bank's confirmation of the transaction an well as a hash/nonce chain linking them with the rest of the log. The whole log entry is signed by the device.

In contrast to the protocol in section 2.5 we do not trust the bank. The messages are all encrypted under a key shared with a bank, but there are also additional message authentication codes computed with a key which is only know by the device and the device manufacturer (K_D). The bank also does not trust the customer, so there are also message authentication codes computed with a key only known by the bank (K_B).

3.3 Security Analysis

The log is designed to be presented to an impartial third party who should be able to verify that it is both accurate and complete. In particular, the customer is producing the log in order to refute a transaction which the bank claims they made. The log entries may be stored by a third party, but they take no part in the log other than that the customer trusts them to store

and accurately reproduce them on demand. In particular, the verifier needs to place no trust in the party storing the logs.

The security properties which are desirable, therefore, are that the consumer cannot forge a log which doesn't contain a real transaction while still appearing valid and that the bank cannot assert that it is not complete when it is.

Looking at these in turn, to create a log which omits a transaction there are several attacks which may be tried. Firstly, can an entry be removed from the log once created. Each entry contains the hash of the previous entry and the hash of a nonce which appears in the next one. If there are any subsequent transactions in the log it will be apparent if an entry has been removed since the chain of hashes will be broken. In addition, the nonce revealed in the later entry won't match the hash earlier committed to.

This leads on to asking whether the next log entry can be massaged so that the missing entry is not noticed. The revealed nonce and the hash of the last transaction could be swapped, but those values are included in the MAC of the log entry. They are also included in the MAC which the bank provides when confirming the transaction. To fake these MACs the attacker would need the secret keys of both the bank and the device.

The next attack is to try and replace a log entry with a mundane one. Again, the next entry in the log would have a hash which did not match and would need to be forged as above. A more interesting attack is to see whether by manipulating the protocol during the protocol run it is possible to immediately create a transaction which follows the last-but-one entry without including the last entry. If it were possible to run the protocol with the bank giving the same nonce-hash and last-transaction hash on a subsequent message then a MAC could be obtained which fits in the chain. However, this would require access to the shared key between the device and the bank. Even in that case, the bank would notice the repeated hashes and be able to produce a duplicate message with a MAC created by the device.

Finally we turn to the attacks by the bank, which is the main way that this protocol differs from that in [38]. The bank takes part in the log protocol by producing a MAC on the hashes forming the log chain in each message. To claim that the customer performed a transaction which is not in the log, they would have to be able to point to the place in the log in which it took place. However, the MAC the bank produced is stored in the log as part of the transaction authorization and certifies the entries either side of it. To claim that the log was not complete they would have to claim that the whole log was fabricated, which would require denying that the customer performed the other transactions. They cannot do so in a court of law.

4 On Internet Shopping

So far the discussion has only been about protecting Internet banking. While this is an important goal, there are many other situations in which financially-sensitive transactions take place and ideally these would also be protected.

There exists a scheme in use by Visa [41] and a similar one by MasterCard [25] where merchant transactions are verified in cooperation with the card issuer. In the current implementation users are redirected to a website run by the card issuer which requests additional authentication details, at the moment a password. This increases the security of general online transactions to that of online banking. As has been seen though, online banking is not that secure. In addition, they are training users to enter authentication details into websites which they are redirected to while shopping. This is something users should be discouraged from doing, not encouraged.

These two systems could be used to integrate a more secure solution, such as the security device proposed here, into online transactions. The infrastructure for hooking into merchant

systems is already there.

5 Conclusion

The current crop of phishing counter measures are either not designed to protect against compromised computers or, if they are, fail to adequately protect. The rising use of mobile phones is also not a panacea. The contribution this paper makes to the area is proposing a trusted UI in a device simple enough to be both cheap to produce and to reason about it's security. The protocols detailed above using the device preserve the security of the system even in the presence of a completely compromised personal computer.

The other innovation is the inclusion of an audit log generated from the transactions. This log is certified by both the device and the bank, the combination of which provides the customer with evidence they can present to refute a phantom transaction that the bank believes was made by them. Existing industry frameworks could be used to integrate this sort of system into other financially sensitive online transactions.

Acknowledgements

We thank the reviewers for their detailed comments and Christian Gehrmann for guiding the paper into its final form.

References

[1] "Specification for the ADVANCED ENCRYPTION STANDARD (AES)". FIPS 197, National Institute of Standards and Technology, Nov 2001. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[2] "Specificatioasn for the SECURE HASH STANDARD". FIPS 180-2, National Institute of Standards and Technology, Aug 2002. http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf.

[3] "Two-Factor Authentication: An essential guide in the fight against Internet fraud". Tech. Rep. GP_WP_2W, GPayments, Feb 2006. http://www.gpayments.com/pdfs/WHITEPAPER_2FA-Fighting_Internet_Fraud.pdf.

[4] Ross Anderson. *Security Engineering*. John Wiley and Sons, jan 2001. ISBN 0-471-38922-6. http://www.cl.cam.ac.uk/~rja14/book.html.

[5] Ross Anderson. "Yet another insecure banking system". *Light Blue Touchpaper*, Oct 2006. http://www.lightbluetouchpaper.org/2006/10/27/yet-another-insecure-banking-system/.

[6] Ross Anderson and Mike Bond. "The Man-in-the-Middle Defence". In "The Fourteenth International Workshop on Security Protocols", April 2006. http://www.cl.cam.ac.uk/~mkb23/research/Man-in-the-Middle-Defence.pdf.

[7] Standard Bank. "OTP FAQ". http://www.standardbank.co.za/SBIC/Frontdoor_02_01/0,2354,3447_13906688_0,00.html.

[8] Neil Chou, Robert Ledesma, Yuka Teraguchi, Dan Boneh and John C. Mitchell. "Client-side defense against web-based identity theft". In "11th Annual Network and Distributed System Security Symposium", February 2004. http://crypto.stanford.edu/SpoofGuard/webspoof.pdf.

[9] Rachna Dhamija and J.D. Tygar. "Phish and HIPs: Human Interactive Proofs to Detect Phishing Attacks". In H. Baird and D. Lopresti (eds.), "Human Interactive Proofs: Second International Workshop", pp. 127–141. Springer, May 2005. http://www.cs.berkeley.edu/~tygar/papers/Phishing/Phish_and_HIPs.pdf.

[10] T. Dierks and C. Allen. "The TLS Protocol". RFC 2246, IETF, Jan 1999. http://www.ietf.org/rfc/rfc2246.txt.

[11] Saar Drimer and Steven J. Murdoch. "Distance Bounding Against Smartcard Relay Attacks". In "Proceedings of the 16th USENIX Security Symposium", USENIX, August 2007. http://www.cl.cam.ac.uk/~sjm217/papers/usenix07bounding.pdf.

[12] Morris Dworkin. "Recommendation for Block Cipher Modes of Operation". Special Publication 800-38A, National Institude of Standards and Technology, December 2001. http://csrc.nist.gov/CryptoToolkit/modes/800-38_Series_Publications/SP800-38A.pdf.

[13] Morris Dworkin. "Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication". Special Publication 800-38B, National Institude of Standards and Technology, May 2005. http://csrc.nist.gov/CryptoToolkit/modes/800-38_Series_Publications/SP800-38B.pdf.

[14] eBay. "eBay Toolbar". http://pages.ebay.com/ebay_toolbar/.

[15] Gemalto. "Press Release", April 2007. http://www.gemalto.com/press/archives/2007/04-18-2007-Barclays.pdf.

[16] Anti-Phishing Working Group. "Phishing Trends Reports". http://www.antiphishing.org/phishReportsArchive.html.

[17] HSBC. "OTP FAQ". http://www.hsbc.com.tr/English/RetailBanking/FAQ/OneTimePassword.asp.

[18] Waterken Inc. "Waterken YURL Trust Management for Humans", 2004. http://www.waterken.com/dev/YURL/Name/.

[19] Markus Jackobsson. "Distributed Phishing Attacks". Cryptology ePrint Archive, Report 2005/091, 2004. http://eprint.iacr.org/2005/091.pdf.

[20] Markus Jakobsson. "Modeling and Preventing Phishing Attacks". In "Financial Cryptography and Data Security", vol. LNCS of *3570/2005*, p. 89. Springer, 2005. http://www.informatics.indiana.edu/markus/papers/phishing_jakobsson.pdf.

[21] John Leyden. "Trusted search software labels fraud site as 'safe"'. *The Register*, September 2005. http://www.theregister.co.uk/2005/09/27/untrusted_search/.

[22] John Leyden. "Phishers rip into two-factor authentication". *The Register*, July 2006. http://www.theregister.co.uk/2006/07/13/2-factor_phishing_attack/.

[23] Virus List. "Virus.WinCE.Duts.a". *Virus Encyclopedia*, July 2004. http://www.viruslist.com/en/viruslist.html?id=1874404.

[24] Virus List. "Worm.SymbOS.Cabir.a". *Virus Encyclopedia*, June 2004. http://www.viruslist.com/en/viruslist.html?id=1689517.

[25] MasterCard. "MasterCard SecureCode". http://www.mastercard.com/securecode/.

[26] MasterCard International. *Chip Authentication Program—Functional Architecture*, Sept 2004. Available upon request from chip_help@mastercard.com.

[27] Trend Micro. "SYMBOS_COMWAR.C". *Virus Encyclopedia*, Oct 2005. http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=SYMBOS_COMWAR.C.

[28] Charlie Miller, Jake Honoroff and Joshua Mason. "Security Evaluation of Apple's iPhone". Tech. rep., Independent Security Evaluators, July 2007. http://www.securityevaluators.com/iphone/exploitingiphone.pdf.

[29] Petros Oikonomakos, Jacques Fournier and Simon Moore. "Implementing Cryptography on TFT Technology for Secure Display Applications". In "The 7th Smart Card Research and Advanced Application IFIP Conference", vol. 3928 of *LNCS*, pp. 32–47. April 2006. http://www.cl.cam.ac.uk/~swm11/research/papers/CARDIS2006.pdf.

[30] OUT-LAW. "Phishing attack targets one-time passwords". *The Register*, Oct 2005. http://www.theregister.co.uk/2005/10/12/outlaw_phishing/.

[31] Bryan Parno, Cynthia Kuo and Adrian Perrig. "Phoolproof Phishing Prevention". In G. Di Crescenzo and A. Rubin (eds.), "Financial Cryptography and Data Security", vol. LNCS of *4107*, pp. 1–19. Springer-Verlag, 2006. http://sparrow.ece.cmu.edu/~adrian/projects/phishing.pdf.

[32] PicStop. "USB Skype Voip LCD Phone—USB-P10D". http://www.picstop.co.uk/Skype-USB-VOIP-Phone/USB-Skype-Voip-LCD-Phone---USB-P10D.

[33] Roberto Di Pietro, Gianluigi Me and Maurizio A. Strangio. "A Two-Factor Mobile Authentication Scheme for Secure Financial Transactions". In "International Conference on Mobile Business", 2005.

[34] Vassilis Prevelakis and Diomidis Spinellis. "The Athens Affair". *IEEE Spectrum*, July 2007. http://www.spectrum.ieee.org/jul07/5280.

[35] QRCode. "About 2D Code". http://www.denso-wave.com/qrcode/aboutqr-e.html.

[36] Blake Ross, Collin Jackson, Nick Miyake, Dan Boneh and John C Mitchell. "Stronger Password Authentication Using Browser Extensions". In "Proceedings of the 14th USENIX Security Symposium", p. 1732. USENIX, 2005. http://crypto.stanford.edu/PwdHash/pwdhash.pdf.

[37] Bruce Schneier. *Practical Cryptography*. John Wiley and Sons, 2003. ISBN 0-471-22357-3. http://www.schneier.com/book-practical.html.

[38] Bruce Schneier and John Kelsey. "Cryptographic support for secure logs on untrusted machines". In "SSYM'98: Proceedings of the 7th conference on USENIX Security Symposium, 1998", pp. 4–4. USENIX Association, Berkeley, CA, USA, 1998. http://www.schneier.com/paper-secure-logs.pdf.

[39] RSA Security. "RSA SecurID Products". http://www.rsasecurity.com/node.asp?id=1311.

[40] USRobotics. "USRobotics USB Internet Phone". http://www.usr.com/products/voip/voip-product.asp?sku=USR9600.

[41] Visa. "Verified By Visa". http://www.visaeurope.com/merchant/handlingvisapayments/cardnotpresent/verifiedbyvisa.jsp.

[42] Candid Wüest. "Phishing in the middle of the stream—Today's threats to online banking". In "The Eighth Association of anti Virus Asia Researchers Conference", March 2006. http://www.hispasec.com/corporate/noticias/recursos/phishing_avar_2005.pdf.

[43] Ye Zishuang and S. Smith. "Trusted Paths for Browsers". In "Proceedings of the 11th USENIX Security Symposium", IEEE Computer Society Press, 2002. http://www.usenix.org/events/sec02/full_papers/ye/ye.pdf.