Mobile Electronic Identity: Securing Payment on Mobile Phones

Chen Bangdao, A.W.Roscoe

Oxford University Computing Laboratory and
James Martin Institute for the Future of Computing
{Bangdao.Chen, Bill.Roscoe}@comlab.ox.ac.uk

Abstract. The pervasive use of mobile phones has created a dynamic computing platform that a large percentage of the population carries routinely. There is a growing trend of integrating mobile phones with electronic identity, giving the phone the ability to prove or support the identity of the owner by containing, for example, a tuple of name, ID, photo and public key. While this helps phone owners prove who they are, it does not prove to them that they are giving their identities to intended parties. This is important in its own right for reasons of privacy and avoiding cases of "identity theft", but all the more important when identity is being provided to support the transfer of value (e.g. in mobile payment) or information. In this paper we show how Human Interactive Security Protocols can support this type of authentication in cases where PKIs are inappropriate, misunderstood or too expensive, concentrating on the case of payment.

1 Introduction

A report from International Telecommunication Union (ITU) earlier this year predicted that there would be 5 billion mobile phone subscribers by the end of 2010 [1]. This number is much larger than the number of personal computers (1,026 million in 2010) predicted by ITU [2]. At the same time, the computing power of mobile phones is ever improving: for example, the HTC Desire mobile phone has a 1 GHz CPU and 576 MB of RAM. In addition to the existing telephony functionalities, mobile phones, especially smart phones, are integrated with various kinds of sensors as well as powerful connectivity, typically on-board camera, GPS, motion sensor, light sensor, Bluetooth, NFC, WiFi, and 3G. Most importantly, they provide well designed convenience for people to use on a daily basis.

Such capabilities have made mobile phones a perfect electronic platform for various implementations. One of the most significant examples of these is the integration of different kinds of Electronic Identities (E-Identities), which helps reduce the number of cards and tokens a person usually carry, for example, ID card, door-access card/token, and bank card or other payment card. Such E-Identities may contain a person's name, photo, fingerprint, public/private keys, or banking/payment account details.

In Japan, the largest mobile operator NTT Docomo began deploying mobile phones containing the FeliCa contactless IC chip in 2004 [3]. The FeliCa contactless chip transforms mobile phones into carriers of various kinds of identities: transportation card, personal ID card and bank card.

It is reported that in 2012, banks and mobile phone operators in the Netherlands will launch a national NFC service which will enable users to use their mobile phones as payment card, tickets, coupons or membership cards [4].

In 2010, Chinese mobile phone operators started to implement a national mobile phone identification policy which requires users to register their mobile phone numbers under their real names and ID numbers. This will create the world's largest mobile phone identification system. At the same time, Chinese banks and mobile phone operators are working together to create a unified national platform for NFC based mobile payment service [5].

Thus there is a huge trend of integrating mobile phones with various kinds of identities, and the most significant use may lies in mobile payment. More generally, we may consider a mobile phone as a bank/payment card once it has logged onto a banking web-site or an e-money web-site like Paypal. Almost all major banks in the US and Europe have opened a mobile banking service.

E-Identities will be communicated between individuals who may or may not know each other, and from individuals to impersonal devices such as doors, merchant tills and web-sites. It is natural to require two things: that you only give your identity to the party that you wanted to give it to, and that you do not accept an identity which you believe attaches to one party when in fact it belongs to another. You may not know in advance the name of the party to whom you are trying to connect.

PKIs are expensive to implement, not usable in cases where the name of the intended connection is not known in advance, and are frequently misused by humans. We need a cheap method of authentication, that allows authentication by context (e.g. that the device you are connecting to is the one in front of you) and which is hard for humans to misuse. We must place into the last category any protocol which simply requires the human user to press a button to say "yes", because particularly in hurried mobile scenarios humans will become distracted and complacent. So while, in mobile-to-mobile connections, it may be a valuable security feature to show each human the photograph of the other, simply expecting them to say "yes" to the obvious enquiry will give only dubious security in practice. In this paper we propose what we think is an appropriate solution to this problem.

To securely transmit an E-Identity, we firstly need to ensure authenticity as well as integrity of the E-Identity, for example, the receiver can trust that the received E-Identity originates from the correct sender. Secondly, we must protect the private E-Identity, no one except from the dedicated sender and receiver can know the details of the transmitted private E-Identity. Thirdly, we have to achieve enough pervasiveness which enables a maximum coverage of mobile phones as well as an implementation of convenient user interfaces.

To satisfy such requirements, we firstly bootstrap an authentic electronic connection between the two parties by using a Human Interactive Security Protocol (HISP), and to fulfill the second requirement, we also bootstrap a session key during the establishment of the connection. In the mean time, a careful selection of an usable HISP can guarantee the satisfaction of the third requirement. Once we have a secure connection, an automatic downloading of such E-Identities is possible, which in some payment processes is made by manually inputting. This can further reduce the amount of human effort.

HISPs are explained in Section 3, which also presents two major mobile payment scenarios; The implementation of the two scenarios is discussed in Section 4, and a general security analysis is given in Section 5.

2 Present-day payment solutions

At present, NFC, Bluetooth and SMS are the main channels used to carry authentication information in payment. Below we review how they are used.

2.1 NFC

NFC is based on a short range (<10cm) RF channel (13.6 MHz), which assumes that the proximity provides sufficient trust of the data transmitted over this channel. NFC is therefore regarded as a typical out-of-band (OOB) channel. OOB channels are sometimes termed as empirical channel or authentic channel, which assumes human trust but allows limited bandwidth of communication. Such channels are common in our daily life, for example, people talking, writing messages, typing words, handshaking, comparing images/words/digits.

An NFC enabled mobile phone can be used as a user-trusted touch point to display and check the received payment amount and the payee's details, as well as confirming the payment. Concrete designs of NFC-based mobile payment can be found in [6, 7]. An NFC enabled mobile phone can act as a card or a terminal, and there is also a mode for peer-to-peer communication and therefore it enables peer-to-peer payment. It gives the convenience of simply touching our mobile phones to communicate securely. We also notice that NFC is currently not widely available among mobile phones, therefore it is not selected in our implementation in Section 4.

However, using proximity as the only authenticator can lead to attacks. For example, a practical NFC relay attack on mobile phones is demonstrated in [8]. In addition, a lack of proper protocols that against man-in-the-middle (MITM) attack may make the implementations of NFC based mobile payment an easier target to MITM attackers [9]. In addition, without link-level security, the transmission between two NFC devices may subject to eavesdropping and data modification [10]. As we were completing this paper there was a press report of a practical MITM attack on a proximity-based car key mechanism [44].

It is desirable that NFC based communication needs to be enhanced by introducing a security protocol that addresses the MITM attack [9]. For example,

we can bootstrap a one-time session key between two NFC devices before transmitting any sensitive data. This key is independent to any existing security and it can be used as an add-on security to NFC.

2.2 Bluetooth

Bluetooth is probably the most popular short-range communication technology available now. According to the Bluetooth Special Interest Group (SIG), in 2014 Bluetooth will be found in 70 percent of all handsets and 83 percent of all netbooks [11]. There are many implementations [12, 13] as well as researches [14, 15]on using Bluetooth in mobile payments.

Bluetooth (v2.0 and older) is known to be subject to searching attack due to its reliance on an arbitrarily human selected passkey [16], and its pairing process generally require a long time which makes it not well user-friendly.

However, the new version Bluetooth v2.1 introduces a Secure Simple Pairing (SSP) scheme which is designed to solve the security problems and falls into the same class of HISPs that we will be studying later in this paper. But this immediately introduces a legacy problem: a communication between a v2.0 mobile phone and a v2.1 mobile phone will be eventually ended as a v2.0 communication.

Any Bluetooth which may fall short of v2.1 is too insecure to support payment. It will be possible to use v2.1 to support the same model of payment we propose in Section 3.

2.3 SMS

Telephony is regarded as a relatively secure communication technology in this paper despite some known attacks [19]. The attacks against telephony network usually require much larger strength in both resources and knowledge, and therefore may not be an "economic" attack against mobile payment. SMS is therefore frequently considered secure. It worries us, however, that this security has no logical basis and is based on purely economic and subjective arguments. Without a formal and provable basis for security it seems unwise to invest heavily in a payment technology.

SMS-based mobile payment methods can be laborious and difficult to learn, and sometimes may not be as instant as other types of mobile payment [17]. The best case for their use may be in long-distance communication in situations where the telephone service providers are able to give a good guarantee of authenticity.

2.4 Other solutions

In [18], the authors discussed an empirical design called MP-Auth which uses mobile phones to protect online banking. Without any use of hardware supports, it is regarded as a typical example of using PKI in mobile payment.

MP-Auth uses two public keys, one is pk_B shared between the PC and the bank, the other is pk_T shared between the mobile phone and the bank. These

public keys are used to bootstrap a symmetric key between the mobile phone and the bank.

In addition, two more procedures are needed: one is to secure the integrity of the data received from the PC, they use an OOB method which is by displaying a hashed result[1] on the mobile phone and the PC, and the user compares and selects the matching one on the mobile phone; the other is to install the correct public key pk_T on the mobile phone, which they recommend to use off-line methods, for example, at a bank branch, through in-branch ATM interfaces, or using telephony.

The use of public keys like this is appropriate in cases such as electronic banking when both parties know it in advance. We do not believe it is otherwise appropriate in the world of ad hoc connections, such as when making a payment to a previously unknown payee.

The solution we will propose can simplify the above processes by considering the two connections between the mobile phone and the bank as a single insecure connection by using an OOB channel between the bank and the mobile phone (see details in Section 3).

Another novel implementation is called Cronto[2]: by using the camera on the mobile phone, the user takes a photo of a square picture similar to a 2-D barcode displayed on his PC screen, and then the device translates the photo into payment details and generates a 6 digits number at the same time, once the user confirms the payment details, he enters the 6 digits number on his PC. By using the camera and the *https* web-site, they create an OOB channel between the user's mobile phone and the bank server. It is considered as a good example of using OOB channels in mobile payment.

3 Using a HISP: mixing context, human trust and security

HISPs achieve what one might at first think impossible: they bootstrap security over insecure networks such as the Internet and WiFi without any pre-existing network of secrets. They do this via the transfer of a small amount of non-secret information, usually by human users, that is authenticated by context.

We hereby assume that in any mobile payment, a payer must have a way of identifying the proposed payee. This identification might arise from already-existing familiarity with the payee or from the context (presence in a shop, in front of a vending machine or through an E-commerce shopping session) in which the need for the payment arises. To understand this better, think of the scenarios in which you would be willing to hand over cash: you might trust a merchant by experience or reputation, you may choose to trust him by context, or you may "trust" him to receive payment because you have already received goods or

[1] They use a correlation function to select the corresponding words to display based on the hashed results

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

services from him. Note that there is a weaker need for trust if, as with handing over cash, you know that the damage that can be caused by an abuse of trust is strictly limited (i.e. to losing a defined amount of cash).

Even when one trusts a large organisation by reputation, one still needs to know that a payment one is making to it is within the payment one thinks one is making.

Some of these means of identification might readily create secure channels: for example one might have retained a channel used for a previous payment to a familiar payee. However some do not, and in some cases there may be a secure channel from a different device (e.g. a browser session on a PC) to the mobile phone from which we want to make payment. However in the great majority of contexts where the need for payment arises, there is an opportunity for the payee to communicate a Short Authentication String (SAS) of 6 digits (say) to the payee in such a way that the payer knows it has come from the intended payee *within the intended payment*. Frequently this will be via an OOB channel such as those formed by the payee looking at a till display or at the *https* window on a browser.

The role of a HISP is to convert well-designed SASs, and the trust that the payee has in the sender, into robust security. An SAS is much more compact than other ways in which one might attempt to authenticate a payee, and much more amenable to incorporation into protocols in a way that is not vulnerable to human mis-use.

To demonstrate our solution, we give two scenarios of mobile phone payment applications:

1. peer-to-peer (phone-to-phone): user A wants to send A's public E-Identity to user B. For example, after verifying A's public E-Identity, B can then make a payment to A[3].
2. customer-to-merchant (phone-to-server): customer C wants to send C's private E-Identity to merchant server M. For example, C uploads C's payment account details to M. This can be an online or a point-of-sale (POS) mobile payment. A mobile phone can connect to the server via: A. a PC; B. telephony or GPRS/3G.

To simplify our discussion, Scenario 2 is discussed in this section, and Scenario 1 is discussed in Section 4.1. In this section, a mobile phone is connected to the server via a PC because this can demonstrate a POS mobile payment as well as an online mobile payment.

3.1 Choosing a HISP

Over the past few years, a new family of authentication protocols that are based on human trust and interaction have been introduced. These protocols are often

[3] This can be completed by sending B's private E-Identity (payment account details) to A, or by sending B's private E-Identity together with A's public E-Identity to a trusted third party, for example, a bank.

referred to as HISPs. They use two kinds of channels: a high bandwidth channel (denoted $\longrightarrow_N$) subject to the Dolev-Yao attack model [29] and a low bandwidth OOB channel (denoted $\longrightarrow_O$). Due to its limited bandwidth, the OOB channel transmits a Short Authentication String (SAS) that is used to authenticate data exchanged over the insecure high bandwidth channel.

By comparing an SAS on an OOB channel, human users can authenticate information received from an insecure high bandwidth channel. Nguyen and Roscoe wrote an extensive survey [28] of HISPs, comparing their cost and efficiency, of which [30, 32, 33] are good examples.

The Symmetric HCBK (SHCBK) protocol [31] is a typical HISP. This, the general description, connects an arbitrary-sized group.

1. $\forall A \longrightarrow_N \forall A' :\ A, INFO'_A, hash(A, hk_A)$
2. $\forall A \longrightarrow_N \forall A' :\ hk_A$
3. users compare $digest(hk^*, \{INFO'_A | A \in G\})$, where hk^* is the XOR of all hk_A's for $A \in G$

SHCBK has each node "publish" its name and a collection of information that it wishes to be authentically connected with that name. It also sends a hash[4] of a randomly generated key hk_A coupled with the name. Once it has received that information from all nodes, and therefore become committed to the set of identities, $INFO$ and hashed keys it will use, it publishes its previously secret hk_A. The point is that by the time of this last publication, it was in fact *committed* to all the data used in the above protocol, even though it does not yet *know* all the hk_As. HCBK stands for Hash Commitment Before Knowledge. A careful security analysis of this protocol (see [31], for example) demonstrates that any attacker is unable to profit from combinatorial analysis aimed at getting the SASs (i.e. digests) to agree even though nodes have difference views of the authenticated information. Good HISPs such as SHCBK therefore offer maximum security for a given amount of human effort.

3.2 Tailoring a HISP

In our payment scenario, only two parties are involved in the payment: customer and merchant. Therefore we have modified SHCBK into a pair-wise protocol which establishes a shared secret key. In the protocol, C represents the mobile phone, M represents a merchant, and U represents a user.

1. $C \longrightarrow_N M :\ ID_C, INFO_C, hash(hk_C, ID_C), hash(k)$
2. $M \longrightarrow_N C :\ ID_M, INFO_M, pk_M, hash(hk_M, ID_M)$
3. $C \longrightarrow_N M :\ \{k\}_{pk_M}, hk_C$
4. $M \longrightarrow_N C :\ hk_M$

5a. $M \longrightarrow_O C : digest(hk_C \oplus hk_M, (ID_C, ID_M, pk_M, k, hash(k), INFO_C, INFO_M))$

[4] Hash means a standard cryptographic hash function that has two main properties: collision resistance, and inversion resistance.

5b. C compares the *digest value*[5] with its own version.

In Messages 1 and 2, we have added 6 more components, k is a session key (a random number) generated by C, it is exchanged by using the uncertified public key pk_M provided by M. To avoid the intruder reflecting hk_C back to C as a supposed hk_M in a way that C would accept, we added ID_M and ID_C as two one-bit tags to distinguish the hashes generated by C and M. $INFO_M, INFO_C$ represent other information that the actual system would require, for example, date and time, part of the payment details, etc.

Naturally, if the protocol has proceeded uninterfered with, C's and M's values will be equal. If, however, an intruder has imposed his own values on the receivers of Messages 1–4, C and M will not agree on all four parameters. For security, what is important is that they agree on pk and k, so we will concentrate on what happens if the intruder interferes with these. What we are concerned about is the chance that the digests agree when these two values do not.

The digest function [30, 31] is designed so that, as hk varies, the probability that $digest(hk, X) = digest(hk, Y)$ for $X \neq Y$ is less than ϵ, where typically ϵ is very close to 2^{-b} for b the number of bits in the output of *digest*. It must also have the property that for any fixed value d, the chance that $digest(hk, X) = d$ as hk varies is less than ϵ. The right value of ϵ is debatable because the larger it is, the more human effort is required. To maintain an acceptable security and usability, implementors need to examine carefully about the use case and the perceived risks between the user and the merchant. A standard [36] given by National Institute of Standards and Technology (NIST) requires that a successful guess of a secret value should be less than one in 1,000,000. Therefore, we put the number of digits of the digest value at 6 in our example[6].

3.3 The human contribution

Depending on human interaction can be dangerous because humans can become lazy, which can disable well designed security. To standardize the work flow of using a HISP, we need to clarify step (5a) and (5b).

In step (5a), when conducting online payments at home, those OOB channels U can directly interact with M are phone calls, SMSs, or using *https* web pages (as most of the banks/merchants are still using *https* service, this does not increase the risks by using it as an OOB channel). Therefore we use a dashed line to show the transmission of the digest value in Fig 1.

[5] The *digest value* represents the SAS that is manually compared by humans.

[6] The SAS here is not secret, but this provides a good analogy. In any case we believe that the use of HISPs in payments should usually be backed up by secondary security as discussed later. 6 digits happens to be the number used in the experiments reported in [35]

To remove the user's complacency[7] in step (5a), we force the user to type the digits of the received digest value into mobile phone[8]. If the comparison of digest value failed at stage (5b), a warning will be displayed on the mobile phone, and we have designed what to do next. In our implementation, we prompt the user to check if he has entered the SAS incorrectly. If so, the protocol is restarted from the beginning. If not, the payment will be aborted, because there is a distinct possibility of the intruder being present.

After a successful run of the protocol, in which C verifies the digest value received from an OOB channel, and at the same time the protocol authenticates the uncertified pk_M and the one-time session key k. The user is convinced that a secure connection is established between him and M.

3.4 Demonstrating a HISP

Once the HISP above has been run, there is a channel between the payer's mobile phone and the payee that the payer trusts as both secret and authentic. We can therefore design payment methods which exploit this high-bandwidth secure channel, thereby increasing the amount of information that can be passed to (a) authenticate the identity of the payer and (b) secure the payment, for example against fraud by the payee.

We give an example of making a payment after successfully bootstrapped the session key k by using a HISP. This largely depends on the actual implementation of banks and merchants.

The session key can now be used to allow secure downloading of payment information from M. U is then asked to approve the payment by password entry. Following this, data necessary to complete the payment can be sent to M over the channel. This will vary depending on the payment protocol being used.

We recommend that an e-cheque is sent, which is encrypted under a bank key (and therefore not understandable by M), together with all information that is not secret from M. This e-cheque might contain M's E-Identity, date and time, amount, $hash(hash(\textit{Payment Info}), \textit{Account Info})$. M sends $hash(\textit{Payment Info})$ to bank. An example protocol is given as below (also see Fig 1):

6. $M \longrightarrow_N C : \{\textit{payment amount, M's E-Identity, date and time, other details}\}_k$
7a. U checks payment amount, merchant's E-Identity, date and time, and other details displayed on C.
7b. If correct, U authorizes the payment by entering password on C.
8. $C \longrightarrow_N M :$ e-cheque

E-cheques provide a way of combatting sophisticated Man-in-the-Shop (MITS) attacks which is discussed in Section 5.3.

[7] A user may simply keep pressing the OK button regardless of what displayed on the mobile phone.

[8] [35] examines ways of performing this comparison and conclusively demonstrates that for security the best approach is for the customer to type the digits of the merchant's digest value into mobile phone, which then compares the two.

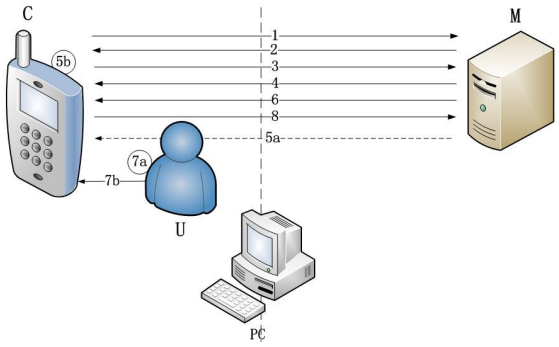

Fig. 1: Using a HISP (demonstration of a successful run).

M can then forward this e-cheque to a bank to get cash.

In each case the fact that the payment details (amount, merchant's E-Identity, date and time) are downloaded onto the mobile phone and approved by the customer gives a considerable secondary security factor over and above that provided by HISP and password.

3.5 Reverse authentication

As we have made clear, the unique feature of a HISP is that it gives the customer confidence that he or she is connected to the desired merchant within the context of the intended payment. This both gives extra security and enables us to make the traditional security goal of authenticating the customer to merchant/bank easier and more thorough. Because it goes in the opposite direction to the main/tranditional authentication accompanying payments, we have termed it *reverse authentication*.

In general, by using reverse authentication, we actually put the users' safety at the center of the security design.

4 Implementation

In demonstration implementations of Scenario 1 and 2 discussed at the beginning of Section 3, we have used the following approaches.

A. Two mobile phones are connected via Bluetooth: the protocol will start after the Bluetooth discover-and-connect process. An e-cheque is sent to the payee from the payer. As explained at the beginning of Section 3, it can be completed in two ways, and to simplify the demonstration, we do not show a second connection to a bank or a third party.
B. A mobile phone is connected to a server: because this can be remote/online or POS payment, we use a PC to act as the display on behalf of the server. To

make the connection instant, the connection between the mobile phone and the server is made by initiating a data call from the server. This is slightly different from the example given in Section 3.4.

4.1 Implementation of approach A

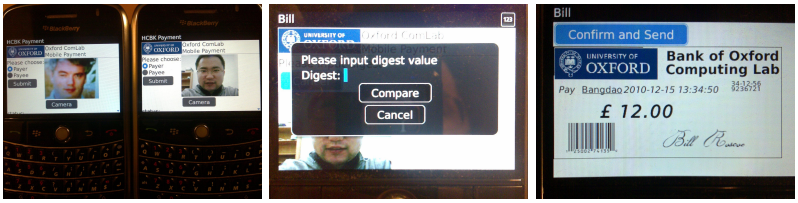

Fig. 2: Peer-to-peer mobile payment implementation.

The photos above show the image of the users, and this is regarded as a useful supplement to the security we discussed in Section 3. By incorporating available biometrics or location information (GPS) into the protocol, we can further enhance the security and provide the user more authentic information to verify each other.

There are two important factors in determining the practicability of this implementation. One is the set-up of connection between two mobile phones, the other is the input of the digest value. In our implementation, the set-up of Bluetooth connection takes around 10 seconds, and the inputing of the 6-digit digest value takes around 15 seconds. However, if one mobile phone can display the Bluetooth address as a 2D barcode, and the other mobile phone reads it by its camera, this can reduce the time of connection set-up. A similar approach can be taken to digest values when this technology is available. This function depends on the performance of specific mobile phones because not all mobile phone cameras can easily film a clear picture of 2D barcode, for example, low-end mobile phone camera can not auto-focus and have difficulty to take clear pictures when hands are shaking. It is, however, an important aspect of our technology that this function can be performed quickly and easily by humans alone.

This is implemented on Nokia N95 and Blackberry 9000: a J2ME Midlet is programmed to run on N95, and a JAVA (on RIM) application is programmed to run on Blackberry 9000. The Bluetooth is v2.0 and the profile is no security.

4.2 Implementation of approach B

In this case, a mobile phone acts as a “trusted device”, which is similar to the current Card Authentication Programme (CAP) readers. And it is required that

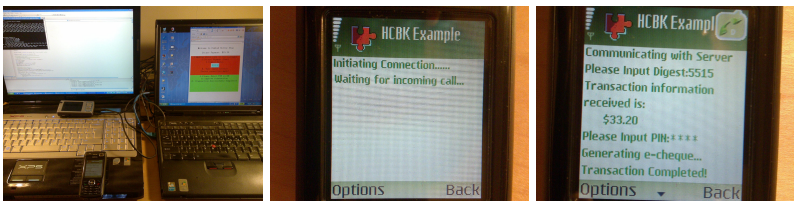

Fig. 3: Customer-to-merchant mobile payment implementation.

the user must activate his or her online banking account or any other payment account before or during the payment process. By using reverse authentication, the merchant's E-Identity together with the payment information is downloaded onto the mobile phone, which can save the human effort of inputting those data required in most current mobile banking applications.

This is implemented on Nokia N70 and a PC (acting as the server): a Symbian C++ application is programmed to run on N70, and a C++ application is programmed to run on the PC.

The cryptography functions we have applied in the applications comply with the guidance published by NIST [37, 38].

5 Security analysis

The security attributes of a mobile payment solution usually include: confidentiality, authentication, integrity, authorization, availability and non-repudiation. Confidentiality and authentication is easily achieved by bootstrapping a secure connection prior the payment process. And the use of strong cryptographic functions protects integrity and can detect any data modification. Authorization is achieved by the verification of: A. user's password; B. user's private E-Identity (banking or payment account details). Non-repudiation is achieved by the use of an e-cheque: a bank will check and verify such an e-cheque, which contains the E-Identities of the two parties as well as the payment details. Availability is not discussed in this paper.

However, except for the above analysis, a few distinct security attributes and state-of-the-art attacks need to be considered carefully.

5.1 Phishing/Credential Harvesting

By means of disguised emails or web pages, attackers lure users to enter their credentials into a fake web form, for example, an online banking log-in form. This is a very common online attack and it is very difficult to defend once the users are tricked into such a web page.

Most mobile payments are immune to such attacks because they have independent applications that handle payment processes: the account details are input locally on the mobile phones rather than on web pages or web forms.

However, without the use of end-to-end security (for example, the use of e-cheque), sophisticated phishing attacks can be developed against mobile payment solutions, for example, a phishing attack can be applied against an NFC based mobile phone by modifying or replacing tags [20]: this can mislead the user to submit data to a wrong party. And some SMS based mobile payment solutions require users to submit their account details in clear text to a third party to log on, and this can lead to an SMS phishing attack: by luring users to submit their account details to a wrong phone number.

Our solution, which provides authentication as well as confidentiality, can ensure that the payer has approved the E-Identity of the payee, and the payee can not reuse anything from the payment. Therefore, it is resistant to the phishing attack.

5.2 Malware

Mobile malware is a serious security threat to all mobile phone based applications. A report from Kaspersky indicates that a total number of 514 pieces of mobile malware have been cataloged between 2006 and 2009 [21]. Such attacks can be detected by installing mobile anti-virus software, for example, Kaspersky, F-Secure and McAfee. And it can be further mitigated by forcing users to download software from the official web sites, for example, Android and iPhone require software to be installed from their official online application shops. However, it may become more difficult to maintain a high level of security with the increasing complexity of mobile phone systems. These issues need to be considered before deciding whether to impose an upper limit on the amount of money allowed in mobile payments. Some discussions about mobile malware can be found in [22, 23].

5.3 Man in the middle

Many NFC based mobile payment solutions are believed to be based on EMV [9, 24], which has been found vulnerable to MITM attacks [25]. And the NFC in itself does lack of a link-level security which may result in eavesdropping, data corruption and data modification [10]. This may make them attractive to MITM attackers.

A HISP, which is designed to be resistant to MITM attack, can protect our solution against any MITM attack (see details in Section 3). However, different implementations may have different set-outs and policies, some MITM attacks need to be carefully examined, for example, the man-in-the-browser (MITB) attack and the MITS attack.

Other types of MITM attack can be found in *https* [41, 42], Bluetooth [43]. [42] shows a more thorough discussion of MITM attacks in tunneled authentication protocols.

Man-in-the-browser attack The MITB attack can be initiated by a MITB trojan embedded in the user's browser, for example, Zeus, Adrenaline, Sinowal and Silent Banker [27], which can then manipulate the online payment session in real-time and carry out legitimate online payments. Therefore, all the solutions that relies on or uses the security provided by web browsers to display payment details on PCs may become vulnerable to MITB attacks.

Defending against MITB attacks can be difficult. For example, the authors of MP-Auth have declared that such attacks are not addressed in their design. And a recent report [26] indicates Zeus trojan is now targeting mobile phones, and it can hijack SMS communication. This will endanger many mobile payment applications that based on SMS or use SMS authentication.

Our solution, which does not depend on any specific connection or display, can resist such attacks by carefully choosing an appropriate OOB channel (see details in Section 3.3). However, the attack on SMS (if successful) does increase the cost of security, for example, we may have to use phone call to deliver the digest value in case of an online/remote payment.

Man-in-the-shop attack The merchant, the one we usually trust, can not guarantee the staff it hires are trustworthy. For example, we can find news like "Don't use cards at petrol stations" [39] or "Restaurant workers indicted in credit card scam" [40]. Same problem arises online – merchant might lose customers' card details or its staff steal data from the server. Various incidents of card data loss are reported on the web [34]. Therefore, users should not give out their card or account details to the merchant because of the MITS attack. Such attacks can be mitigated by using the concept of e-cheque which is discussed in Section 3.4. Or the payment may has to be made by a trusted third party: a bank or a mobile wallet service provider. And the merchant will be informed and invoiced by the trusted third party.

6 Conclusion

We have demonstrated that using a HISP on a mobile phone can help the customer to create a secure connection which "reversely authenticates" the merchant (the payee), while keeping a low-cost on human's effort. This solution helped by the flexibility of an OOB channel which assumes no existing security can be used to defeat MITM attacks as well as to allow an efficient and secure transmission of E-Identities. And the discussion of ϵ would be useful – the balance between security and usability, which can provide more guidance to future implementations of online payment solutions based on HISPs.

Acknowledgement

This project was funded in part by grants from US Office of Naval Research and the Oxford Martin School. We would like to thank Long Nguyen and Ronald Kainda for their contribution to the background of this work.

References

1. ITU Report. *ITU sees 5 billion mobile subscriptions globally in 2010.* http://www.itu.int/net/pressoffice/press_releases/2010/06.aspx
2. ITU Report. *Personal Computers market.* http://www.areppim.com/stats/stats_pcxfcst.htm
3. L. Srivastava. *Japan's ubiquitous mobile information society.* In J. Policy, Regulation and Strategy for Telecommunications 6(4), 2004.
4. Reuters. *Dutch deal paves way for mobile payments in 2012.* http://uk.reuters.com/article/idUKLDE6880OC20100909
5. Finextra. *China Telecom, Bank of China and China UnionPay launch mobile proximity payments.* http://www.finextra.com/news/announcement.aspx?pressreleaseid=36776
6. M. Pasquet, J. Reynaud, and C. Rosenberger. *Secure payment with NFC mobile phone in the smarttouch project.* In Symposium on Collaborative Technologies and Systems, 2008.
7. K.S. Kadambi, J. Li, A.H. Karp. *Near-field communication-based secure mobile payment service* In Proc. the 11th International Conference on Electronic Commerce, 2009.
8. L. Francis, G.P. Hancke, K.E. Mayes, and K. Markantonakis. *Practical NFC Peer-to-Peer Relay Attack using Mobile Phones.* In Workshop on RFID Security, 2010.
9. R. Anderson, *RFID and the Middleman.* In Proc. Financial Cryptography and Data Security, 2007.
10. E. Haselsteiner and K. Breitfuss. *Security in Near Field Communication.* In Proc. Workshop on RFID Security, 2006.
11. Bluetooth SIG. *SPECIAL REPORT, Quarter 4, 2010.* http://signature.bluetooth.com/bluetoothsig/2010Q4?pg=22#pg22
12. J.J. Chen, C. Adams. *Short-range wireless technologies with mobile payments systems.* In Proc. the 6th international conference on Electronic commerce, 2004.
13. S. Pradhan, E. Lawrence, A. Zmijewska. *Bluetooth as an Enabling Technology in Mobile Transactions.* In Int'l Conference on Info. Tech.: Coding and Computing, 2005.
14. K. Zolfaghar, S. Mohammadi. *Securing Bluetooth-based payment system using honeypot.* In Int'l Conference on Innovations in Info. Tech., 2009.
15. J. Gao, K. Edunuru, J. Cai, S. Shim. *P2P-Paid: A Peer-to-Peer Wireless Payment System.* In Proc. WMCS '05.
16. M. Jakobsson, S. Wetzel. *Security weaknesses in Bluetooth.* In Proc. CT-RSA 2001, LNCS, vol. 2020/2001, pp. 176-191.
17. N. Mallat. *Exploring Consumer Adoption of Mobile Payments - A Qualitative Study.* In J. Strategic Information Systems 16(4)(2007), pp. 413-432.
18. M. Mannan and P.C. van Oorschot. *Using a Personal Device to Strengthen Password Authentication from an Untrusted Computer.* In Proc. Financial Cryptography and Data Security, 2008.
19. C. Mune, R. Gassira, R. Piccirillo. *Hijacking Mobile Data Connections.* http://www.blackhat.com/presentations/bh-europe-09/Gassira_Piccirillo/BlackHat-Europe-2009-Gassira-Piccirillo-Hijacking-Mobile-Data-Connections-whitepaper.pdf
20. G. Madlmayr, J. Langer, C. Kantner, J. Scharinger. *NFC Devices: Security and Privacy.* In Third Int'l Conference on Availability, Reliability and Security, 2008.

21. A. Gotstev, D. Maslennikov. *Mobile Malware Evolution: An Overview, Part 3.* http://www.securelist.com/en/analysis/204792080/Mobile_Malware_Evolution_An_Overview_Part_3
22. G. Lawton. *Is It Finally Time to Worry about Mobile Malware?* In J. Computer, 41(5):12 - 14, 2008.
23. C. Fleizach, M. Liljenstam, P. Johansson, G.M. Voelker, A. Mehes. *Can you infect me now?: malware propagation in mobile phone networks.* In Proc. WORM '07.
24. R. Sanders. *From EMV to NFC: the contactless trail?* In J. Card Technology Today, vol. 20(3), 2008.
25. B. Adida, M. Bond, J. Clulow, A. Lin, S. Murdoch, R.J. Anderson and R. Rivest. *Phish and Chips.* In Security Protocols Workshop, 2006.
26. S21sec. *ZeuS Mitmo: Man-in-the-mobile.* http://securityblog.s21sec.com/2010/09/zeus-mitmo-man-in-mobile-i.html
27. RSA Lab. *Making Sense of Man-in-the-browser Attacks.* http://www.rsa.com/products/consumer/whitepapers/10459_MITB_WP_0510.pdf
28. L.H. Nguyen, A.W. Roscoe. *Authentication protocols based on low-bandwidth unspoofable channels: a comparative survey.* In J. Computer Security, 2010.
29. D. Dolev and A. Yao. *On the security of public key protocols.* In Information Theory, IEEE Transactions on, vol. 29(2), pp. 198-208, 1983.
30. L.H. Nguyen and A.W. Roscoe. *Efficient group authentication protocol based on human interaction.* In Proc. FCS-ARSPA, 2006.
31. L.H. Nguyen and A.W. Roscoe. *Authenticating ad hoc networks by comparison of short digests.* In J. Information and Computation 206 (2008).
32. S. Vaudenay. *Secure Communications over Insecure Channels based on Short Authenticated Strings.* In Advances in Cryptology - Crypto 2005, LNCS vol. 3621, pp. 309-326.
33. S. Laur and K. Nyberg. *Efficient Mutual Data Authentication Using Manually Authenticated Strings.* In Proc. Cryptology and Network Security, 2006.
34. Dataloss. http://datalossdb.org/search?query=card
35. R. Kainda, I. Flechais and A.W. Roscoe. *Usability and Security of Out-Of-Band Channels in Secure Device Pairing Protocols.* In Proc. SOUPS, 2009.
36. NIST. *Security Requirement for Cryptographic Modules.* FIPS 140-2, 2002.
37. NIST. *Recommendation for Key Management.* SP 800-57, 2007.
38. NIST. *Cryptographic Algorithms and Key Sizes for Personal Identity Verification.* SP 800-78, 2010.
39. Times Online. *Don't use cards at petrol stations.* http://www.timesonline.co.uk/tol/money/consumer_affairs/article1400176.ece
40. Startribune. *Metro restaurant workers indicted in credit card scam.* http://www.startribune.com/local/west/102029153.html
41. F. Callegati, W. Cerroni and M. Ramilli. *Man-in-the-Middle Attack to the HTTPS Protocol.* In IEEE Security & Privacy, 2009.
42. N. Asokan, V. Niemi, and K. Nyberg. *Man-in-the-Middle in Tunnelled Authentication Protocols.* In Security Protocols Workshop, 2005.
43. D. Kugler. *"Man in the Middle" Attacks on Bluetooth.* In Proc. Financial Cryptography, 2003.
44. D. Tobin. *Open sesame: the magic car thieves.* The Sunday Times, 6 Feb. 2011.