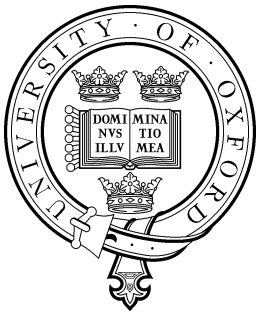
Using Human Interactive Security Protocols to Secure Payments



Bangdao Chen

Keble College

Oxford University Computer Science Department

Doctor of Philosophy

September 2012

This thesis is dedicated to my lovely baby daughter Sitong.

Table of Contents

List of Figures

List of Tables

Abstract

We investigate using Human Interactive Security Protocols (HISPs) to secure payments. We start our research by conducting extensive investigations into the payment industry. After interacting with different payment companies and banks, we present two case studies: online payment and mobile payment. We show how to adapt HISPs for payments by establishing the *reverse authentication* method. In order to properly and thoroughly evaluate different payment examples, we establish two attack models which cover the most commonly seen attacks against payments. We then present our own payment solutions which aim at solving the most urgent security threats revealed in our case studies. Demonstration implementations are also made to show our advantages. In the end we show how to extend the use of HISPs into other domains.

Acknowledgement

The research presented in this thesis is supported by many people. I must first thank my supervisor Professor Bill Roscoe who did not only provide me guidance and tutorials but also teach me skills of research and positive thinking. He encouraged me to actively and correctly conduct our practical research in payments. He helped me to overcome many barriers of getting necessary resources for our research.

Our research in payment is an on-going project operated by ISIS-Innovation Ltd. I must thank project manager Emma Sceats, Brendan Spillane, Roy Azoulay and Andy Robertson who managed our project in the past five years. They helped to establish contacts with banks and payment companies and organised many meetings with them. I also thank Dr. Wenming Ji and Dr. David Baghurst who helped to prepare my visit to the Chinese payment industry in 2011.

My visit to the Chinese payment industry was funded by ISIS and was organised by CSIP (National Software and Integrated Circuit Public Service Platform) in China. I must thank Mr. Gao Songtao, director of CSIP, who introduced our research to many Chinese banks and payment companies. I must thank Mr. Wang Lixun, manager of CSIP, who accompanied me to attend a wide range of meetings during the hot summer in Beijing and Shanghai in 2011. I also thank Mr. Liu Longgen, Ms. Tao Yingying from CSIP for their help of preparing meetings and documents.

I thank Dr. Long Nguyen for his efforts of developing protocols that are used in our research. I thank Dr. Ronald Kainda for his work on the usability of the protocols we used. I thank Mr. Huang Xin for his work on sensors. I also thank Dr. Ivan Flechais who offered much help during our recent research.

In the end, I want to thank my parents who offered their great love and support.

I want to thank my wife Jackie who suffered a lot and endured my absence during my studying in Oxford. I want to say thank you to my new born baby daughter, who missed her father since she was born.

Chapter 1

Introduction

1.1 Motivation

Ad-hoc networks are frequently used to facilitate various electronic transactions in our daily life; for example, online and mobile payments. The need for ad-hoc network security has been substantially diversified thanks to the growing usage of pervasive mobile electronic devices, such as smart mobile phones and tablet computers. These pervasive devices allow us to make connections to online services in any place at any time. Security for ad-hoc networks should, therefore, be flexible and strong in order to adapt to changing environments and increasing security requirements. This thesis concentrates on cases where we need to connect devices securely to each other.

One of the most valuable applications, and one that is particularly attractive to attackers, is payment. Electronic payments require well-designed security protection and infrastructure support, because they need to transfer information securely and authentically. Electronic identities and physical tokens are frequently used to authenticate and protect electronic payments. However, vulnerabilities can be found, many of which have been exploited in order to carry out various attacks to harvest credentials or steal money.

The payment environment has been further complicated by the growing trend for implementing electronic payment solutions on mobile phones. Their mobility and coverage of day-to-day payments provides unprecedented convenience to users, but also new opportunities for attackers. In recent years, sophisticated attack techniques have been developed, and new attack vectors have emerged. We concentrate on the connection of a mobile device to a second device in order to make a payment, where a pre-existing insecure connection already exists or can readily be created.

Traditional security solutions, such as shared secrets or PKI, become inconvenient in these cases. For example, the customer meets a merchant for the first time (online or offline), and the customer wants to pay the merchant by sending an electronic cheque. The customer first authenticates the connection established between its device and the merchant's device; if it is correct, then using a key to encrypt the e-cheque before sending it to the merchant. In this case, there are no existing shared

secrets (e.g. passwords or shared keys) between them to encrypt the connection; and methods that use PKI, for example, the secure socket layer (SSL) protocol, may not be able to guarantee robust authentication since the customer may not know the merchant's complete name that is included in the public key certificate, and hence they cannot ensure the secrecy of the e-cheque. We conclude these problems as following:

- Humans often choose weak passwords which are subject to brute-force searching attacks. And humans easily forget passwords especially when the number of passwords is large. In addition, because passwords are static, they are also vulnerable to phishing and key-logging attacks [79]. Passwords can only be used in the context of a pre-existing security infrastructure or where one has the time and patience to set them up.

- A public key certificate binds a name to a public key, but it will become less useful when the user does not know his or her counterpart's name, for example, in a payment between two strangers. Such bindings need to include more context to become meaningful to humans.

Furthermore, when using online services, we have to trust the service providers and/or third parties. However, our personal interests often contradict their business interests: a company, especially when its size is small, may not have the proper security protection for our private data, or an insider can easily steal our data from their databases. This could happen, for example, when we make a payment at a small shop, or enjoy a location-based game provided by a new social network company. A report from Verizon [34] indicates that the number of data breach incidents increased dramatically in 2010.

A technology that helps to overcome such problems is Human Interactive Security Protocols, or HISPs. As the name suggests, these allow the human users to participate in a meaningful way, and they permit the authentication of parties by the contexts in which they sit, rather than just by identity. We will discuss these ideas extensively in this thesis.

This thesis is designed to examine the potential use of HISPs in payments. We focus on solving challenges found in traditional and new payment applications and services, such as, online payments and mobile payments. We also present a few new challenges and research problems discovered during the implementations of these solutions.

1.2 Human Interactive Security Protocols

Over the past few years a number of what are sometimes termed "Human interactive security protocols" [97, 100, 119, 120, 121, 123, 122, 138, 154], or HISPs, have been developed that permit one or more humans to bootstrap strong security between two or more devices based on the non-fakeable transmission of a minimal quantity

of data between them to supplement a normal insecure communications medium. Because the humans know which systems they have communicated this data between (typically a few characters long and which we will refer to as a *short authentication string* (SAS)) they know which systems are connected about:startpagesecurely. There is an important difference between these protocols and those that bootstrap security from passwords; namely that the SAS does not have to be secret.

This class of protocols allows two or more parties who trust one another, or a single party who trusts one or more others, to bootstrap a secure network using no more than an ability to communicate a small number of bits over a human-based, non-fakeable channel. Another way of looking at them is that if the human(s) involved create an insecure channel between their devices, and already have an unfakable way of passing a small amount of information amongst them, then they can either turn the insecure channel into a secure one or discover the presence of an intruder who is trying to subvert it.

The best of these protocols, for example those in [97, 100, 119, 120, 121, 123, 122, 138, 154], enable assurance to these humans that there is no attack that would allow an intruder to get the system into an insecure state (where the connections established are other than what the humans believe), with probability meaningfully greater than 2^{-b}, where b is the number of bits in the SAS. In addition, to have such a chance, the attacker will have a $1 - 2^{-b}$ chance of his presence being revealed by the difference between the strings. In particular, these protocols prevent any combinatorial searching by the intruder improving its chance of success.

They thus provide a convenient way to bootstrap security that can be used in a wide variety of ways in contexts both where all the devices are co-located and where they are not, and where the authentication is provided to all devices or asymmetrically to one, because only that device's user has observed the equality required of the SASs. Similarly, they can be used in convenient consumer devices or as part of the security process in a more elaborate type of system.

This is a new approach to security and requires novel approaches at multiple levels. Details of the protocol we will use are discussed in Section 3.4.

1.2.1 Context

A HISP allows humans to bootstrap security based on contexts. Contexts here represent facts humans used to cultivate trust between each other, for instance, locations, environments, texts, images, audios, memories, experiences and other facts that can be perceived by humans. For example, a customer walks into a shop and makes a payment to the shop keeper. In this case, the location – both the customer and the shop keeper are inside the shop – gives the most distinct fact (perhaps supplemented with the fact that the customer has bought something) that the customer and the shop keeper are the payer and the payee; and the visual and audible communication – both the customer and the shop keeper can see and hear from each other – gives the most usable fact of that there is a non-fakable channel between the customer and

the shop keeper.

1.2.2 Empirical channel

We have stated that a HISP uses an unfakeable channel on which the SASs are compared. In practice, this channel is often called an *out-of-band* (OOB) channel or an *authentication channel*. In this thesis, we also call it an *empirical channel*. There is no assumption that communication on this channel is secret. In order to reduce ambiguity, we use the term empirical channel in most discussions. In our case studies, we also use the term OOB channel when necessary; for example, in payments, many companies claim that the use of telephony is an OOB channel.

An empirical channel is usually a human-based channel, for example, face-to-face communication, voice call and video call. Some online communication channels also fall into this category. For example, as long as the user trusts, SSL connections, social network pages[1] can be used as empirical channels. The condition is that the user trusts that the channel is established between him/her and the correct instance of his/her counterpart. This is confirmed through the acceptance of the key certificate or the acknowledgement of the association between the social network page and the person.

The above discussion does not include all possible empirical channels. In general, a communication channel C can be used as an empirical channel if it can satisfy the following requirements: (i) it is infeasible to attack the authenticity of the data sent over C; (ii) it is infeasible to attack the integrity of the data sent over C. To quantify the infeasibility of the two requirements, we use PR_C to represent the probability of failing any one of the two requirements given the customer has accepted the value received via this channel. Some conditions are defined as following:

1. L represents the amount of information (in bits) that is sent and compared over C.

2. PR_S represents the probability of a successful secure connection given that the customer believes one has been made in a certain payment scenario.

We then define that channel C can be used as an empirical channel in scenario S if C can satisfy the following criteria:

$$1 - PR_C - 1/2^L \geq PR_S$$

We call it the *Empirical Evaluation Criteria*. An interesting fact of this criteria is that we must consider the scenario where the empirical channel is used. If the scenario changes, the empirical channel is likely to change as well. For example, when making a payment, the amount of the payment clearly affects the security

[1] A social network page needs to have SSL protection before the consideration of whether or not to use it as an empirical channel.

requirement: the higher the payment amount is, the stronger the security the system requires. Consider the case where both SSL channel and telephone voice channel are available, assume that SSL channel is less secure than telephone voice channel[2], when designing a system to allow large amount payment (e.g. greater than 10000 US dollars), a better option is to use the telephone voice channel as the empirical channel; when designing a system to allow small amount payments (e.g. less than 100 US dollars), the SSL channel may be a better choice since it is cheaper. Or if the variation of PR_S is small, for example, payments between a few dollars to tens of dollars, we may simply increase or decrease L and keep using the same empirical channel.

Another interesting fact is that there can be many methods to compute PR_C. For example, it is possible for an attacker to impersonate others both in the physical world and the online virtual world. We have mentioned that context is used so that humans can quickly make their own decision of whether or not to "trust" the data received via the empirical channel. But how to accurately quantify trust? And how to thoroughly evaluate context? These two questions are difficult and there are likely no simple methods or rules to follow.

Also a hidden fact is that L cannot grow longer than a certain value, say α. α is the value which does not only satisfy the Empirical Evaluation Criteria but also allows humans to conveniently and correctly compare the SASs. There can be many ways to compare the digest value, for example, except for numerical input, people can read and compare it, or we can display the digest value as words or pictures, in a way that people can quickly and correctly compare. Research such as [44, 46, 70, 90, 89, 106, 107, 141, 145] provides good resources of evaluating human factors. We therefore refine the Empirical Evaluation Criteria into the following one:

$$1 - PR_C - 1/2^L \geq PR_S \quad \text{AND} \quad \alpha \geq L$$

Note that the above criteria only provides a general guidance of evaluating empirical channels. To make it computable, a designer needs to first give a scale of values and then to assign values according to evaluation results of different parameters. In the following sections we will present some examples of evaluating PR_C and α.

• Evaluating PR_C

We have discussed that there are no simple methods or rules to quantify trust and evaluate context. In order to simplify the security analysis of our payment solution presented later in this thesis, we introduce some guidelines of evaluating the security of a few empirical channels we have observed in this section.

The definition of trust from the *Oxford Dictionary of English* (third edition) is: firm belief in the reliability, truth, or ability of someone or something. In this thesis

[2]The assumption may not stand in practice. For example, the telephone voice channel is less useful if the two parties cannot recognise each other's voice or the phone number.

trust means the payer's cognition of the fact of whom is the payee and the fact of receiving information from the payee. Human cognition is a very complicated subject. Clearly it is not our intention to study this topic in our thesis. We only need to discuss the risks of using different empirical channels. In other words, we only focus on methods that an attacker can use to "fool" the payer into believing that the digest value received from the attacker is from the payee.

According to the Model Human Processor [47], a human can be considered as an information processor which takes inputs of visual and audio stimulus[3] and generates outputs of actions (e.g. pressing a button). A cognitive system is used to connect inputs perceived by humans to the right outputs of actions. It works by using the memory or the knowledge we have already established to process the inputs.

This leads to our method of evaluation. We will analyse the physical stimulus and the human knowledge involved in using one empirical channel. Our discussion in this section only covers a few selected empirical channels, for example, face-to-face communication, phone call, SMS, VoIP voice call, communication on social networks, and communication via *https* web-pages. Table A.1 in Appendix A shows the type of stimulus and the potential knowledge of the payer involved in using a certain empirical channel. The score assigned to each empirical channel is the security score. The notes contain our explanation of giving the score.

Note that this evaluation only provides estimations of the security qualities of the selected empirical channels. The scores we give are meaningful when comparing empirical channels. In other words, the scores are useful when selecting an empirical channel to use. A score alone is not sufficient to determine whether or not an empirical channel can be used in a certain payment scenario.

This is because it is sometimes very difficult to determine the context of an empirical channel. And when the context changes, the condition of using an empirical changes as well. We give two examples as following:

- Face-to-face communication. It is difficult to determine how well the payer knows that this is person he/she wants to pay. For example, when we say the payer knows the payee, it could mean they know each other well or it could be that the payer has only limited knowledge or vague impression about the payee. But in either case, face-to-face communication is better than most other empirical channels.

- Phone call or voice call. The quality of voice is determined by many factors. When the quality of voice is low, the payer may not be able to tell whether the voice belongs to the right payee or not. Similar problem can be found in video calls.

We recommend that designers may only consider the use of an empirical channel with score above or equal to Medium-. In order to simplify our discussion, some

[3]In order to simplify our discussion we do not discuss tactile sensation in this thesis.

conditions are not discussed. For example, human voices can be synthesized; videos can be animated or edited and reused; and human faces can be faked using special techniques.

There are many empirical channels that are not discussed here. For example, in face-to-face communications, the payee may be represented by a till machine. In this case, its location and perhaps some signs or logos help the payer to recognise it from others. Postal mail can be used as an empirical channel, but its speed of communication may not suit the requirement of payments. Emails secured by using digital signatures can be used as an empirical channel, but this will require the payer to install certain certificates and software to correctly verify digital signatures. We are unable to enumerate all possible empirical channels in this thesis. A general guidance is that before using an empirical channel in a certain scenario, the designer needs to carry out an investigation of its security and usability within the given scenario.

• Evaluating α

The value of α can be used to indicate the usability of an empirical channel. For example, the more the amount of information is allowed to be exchanged and compared over an empirical channel conveniently and correctly, the higher the usability the empirical channel has. We are not aware of any experiments dedicated in searching for the maximum value of α over empirical channels. But there are various usability tests of different comparison methods over empirical channels. We will give a general discussion of these methods in this section.

There are usually two directions of comparing SASs over an empirical channel: (i) comparison by human; (ii) comparison by machine. The first direction is perhaps the most intuitive way of comparing SASs. For example, the payer first reads the SAS displayed on the payee's device and then compares it with the one displayed on his/her own device; if it matches, the payer presses the confirm button. The authors of [90] tested 14 different methods of comparing a 20-bit long value. Among these the methods of comparing numeric or alphanumeric values are the most efficient in terms of completion time. The authors of [153] made tests using 4-digit, 6-digit and 8-digit decimal values. They indicated that 8-digit values were considered to be hard to use.

The second direction, when it allows, can significantly increase the value of α. For example, the SAS can be displayed as a 2D Barcode on the payee's device, the payer then use his/her device's camera to take a picture of the Barcode; the Barcode is then translated into the value of the SAS and compared with the version generated by the device automatically. However, to use this method conveniently, many factors need to be considered, for example, the definition of the screen where the Barcode is displayed, the ambient light, the quality of the payer's camera, and the steadiness of the payer's hand holding the camera. [107] shows an evaluation of this method. Other methods involve LED flash [142], sound [70] and infrared [37].

- **Uncaptured properties**

Except for security and usability, other properties of empirical channels may affect their use, for instance, availability and cost.

Availability shows the limit of some empirical channels. For example, face-to-face communication may not be available when the two people are far away from each other. Deaf people cannot use any voice channel without special help.

Cost affects the designer's decision of whether or not to use a certain empirical channel in a certain scenario. For example, telephony can sometimes be expensive and it is therefore may not suit small payments.

1.2.3 Finding an empirical channel

One condition of using HISPs in payments is to find a usable empirical channel. Our assumption is that it is always possible to find an empirical channel in any human-initiated payment. This is based on our observation: we never pay someone without a reason. The reason is the cause of the payment or the context of the payment.

Based on our observation, there is always some kind of human-based communications in a payment to establish that reason. Actions like buying goods in a shop, buying services or goods from a person, or buying goods online, all involve human communication. For example, when we go into a shop, choose the goods, and go to the till, such movements constitute our communication with the shop as does observing the shop and the cash till. Other examples include making phone calls, sending text messages, video or audio interactions, Instant Messaging, social networks, and emails, and including, of course, face-to-face communication.

Based on the above analysis, we understand that before any payment has taken place, there must be some interaction via human communication, which tells the human whom to pay and justifies the cause of payment. Such communication is an integral part of how we identify the party that we pay. To identify the payee properly, the payer must have a high degree of confidence that the identifying information/empirical channel is actually from the intended payee. This leads to our assumption presented earlier in this section.

However, an empirical channel found in a payment scenario needs to be evaluated using the Empirical Evaluation Criteria presented in the previous section. If the empirical channel found does not satisfy the criteria of the scenario, then it cannot be used. In this thesis we only focus on examples where empirical channels can be found and used.

1.3 Objectives

Our main objective is to find out how to use HISPs to develop flexible and efficient solutions of making secure payments. To achieve this end, we design our methods as follow:

1. Determine and evaluate the attributes of payments;
2. Establish a method of authentication in payments;
3. Analyse attacks against payments and evaluate existing payment solutions;
4. Design and develop payment solutions using HISPs.

1.4 Research methods

A challenge to payment research is the lack of resources. Most documents specifying payment systems are not publicly available. Academics often have to reverse engineer payment systems to understand how they are functioning. For example, researchers at Cambridge University have reverse engineered the Chip Authentication Program card reader, which enables them to examine and reveal its vulnerabilities [61].

Current academic research on payments is often limited to theory. Others provide empirical models of implementations [160, 36, 29, 103, 77, 35, 140, 67] but few have been tested in practice. The limited communication between academic research and industry implementation restricts research on solving real-life problems in payments.

The relevance of the research in this thesis has been ensured by the following: (i) close attention to the available literature as it becomes public; (ii) meetings were held with a range of leading banks and payment technology companies in the UK and China; (iii) confidential documents of payment systems were obtained through industry support.

Substantial efforts have been made to gain access into the "inside" of the payment industry. It is worth mentioning that an extensive investigation in the Chinese payment industry was made between August and September in 2011. Through these efforts, we gathered and evaluated resources that allowed us to identify the most urgent problems of the existing payment market. Our research, therefore, aims at providing solutions that can be used to provide secure and convenient payment services in practice.

1.5 Main contributions

1.5.1 Framework of using HISPs in payments

We have established a method of authentication in payments:

> In transactions involving Alice proving her identity to some party Bob, whom she can identify by context, it often makes sense for her to get a connection that she knows is with Bob, even if she does not know Bob's name. She can then use that connection to prove her identity securely, and perhaps perform other functions, so that she has no need to place a credit

> card or identity card etc., in the hands of another party, thereby enabling her control over what information is taken. In other words, before she authenticates herself in one direction, she performs an authentication in the *reverse* direction. We call it *reverse authentication*.

We build our payment solutions based on this method. In general, our payment solutions complete three main steps:

1. The payer establishes an insecure electronic connection to the payee.
2. The payer authenticates and secure this connection via a HISP. The authentication here is to authenticate the public key together with other necessary information received from the payee. If the public key is authenticated, it will be used to establish a shared secret key to encrypt the connection.
3. The payer and the payee exchange necessary information to complete the payment.

In Step 1, we assume all electronic connections we use in payments are initially insecure. This is a crucial assumption which simplifies the structure of communication. For example, the connection between the payer and the payee may consist of multiple connections connecting various routers and intermediate parties, which can be considered as a single insecure connection under our assumption.

In Step 2, we assume that we can find or create an empirical channel in all human-initiated payment scenarios. An evaluation is needed to check whether the empirical channel can satisfy the security requirements or not.

In Step 3, we assume there is an underlying payment system supporting the payment over the secure connection established between the payer and the payee. For example, the payee first sends the order information[4] to the payer who can verify it and then issue an electronic cheque (the encrypted payment information[5]) to be sent to the payer (or to a bank or a third party). We will discuss this in detail in Chapter 6.

The method of reverse authentication, the three main steps and the three assumptions constitute the framework of using HISPs in payments.

1.5.2 Advantages of using HISPs in payments

Advantage 0 A HISP can allow the user to store/access payment account details, card details or other credentials locally on his/her own device to keep them private;

[4]The order information normally contains the name of the good or the service purchased by the payer, the amount of payment, the account details of the payee, the time and date of the payment, and other information required by the payment system.

[5]The payment information normally contains the order information, the card details and account details of the payer, and other private information (e.g. password) of the payer required by the payment system in order to authorise the payment.

it can allow the user to download the order information from the merchant (the payee) automatically; and the user can securely transmit this sensitive information electronically. The secure electronic connection allows a larger amount of information to be exchanged between the payee and the payer, thereby allowing enhanced and more secure payments.

Advantage 0 is the most significant advantage of all. Firstly, the user can keep his/her payment information private. Secondly, he/she no longer needs to type the order information on his/her device. Thirdly, he/she can read and verify order information downloaded automatically from the merchant on the display screen of his/her own trusted device. Fourthly, he/she can then send an electronic cheque to the merchant or the bank (or a payment company), which is a more secure and convenient way to complete a payment.

In addition, we also reveal the following advantages that are essential to be noticed during implementation :

Advantage 1 HISPs can help the regulation of human behaviours in payments.

Advantage 2 When establishing the initial electronic connection in payments, HISPs can facilitate this process by allowing the payer to use contextual information about the payee.

Advantage 3 When using context in payments, HISPs can help reduce the complexity of the metrics of context to be captured and measured.

We will demonstrate all of the above in this thesis.

Advantage 1 is necessary because humans can be complacent. Humans may, for instance, ignore important warnings and skip necessary security checks if they are allowed to. In general, human complacency can disable well-designed security. This problem can be worse in payments because humans may need to make many payments a day. Requirements such as properly verifying the merchant's account number and name may be difficult to satisfy because: (A) the user may not be able to compare these details accurately; (B) the user may easily skip this check because he/she is complacent; (C) the user may lack information about the merchant. By using HISPs well, we can eliminate human complacency: we force the user to manually input the digest value. And we can at least achieve the equivalent security of cash payment. We discuss this in detail in Chapter 3.

Advantage 2 is useful because the initial establishment of the electronic connection largely determines the spectrum of where the payment solution can be used, and the usability of the payment solution as well. In general, we have to find a way to conveniently bridge the relation between two machines and the relation between two humans. In other words, we have to find a solution to the following challenge: how can we securely establish an electronic connection between two devices with minimum cost of human effort? HISPs does not rely on any existing security. Therefore, the

connection can be initially insecure. This provides more flexibility and freedom to choose which connection to use, as well as how to establish it.

Advantage 3 is beneficial when developing novel payment solutions based on contexts. Defining what is the proper context for each application can be difficult, and some contexts are difficult to capture such as human trust. By using HISPs, this problem can be generalised as the user's choice whether or not to trust the data received from the empirical channel.

In addition, the security of relations established via contexts is difficult to prove. For example, it is difficult to verify the relation between the payer and the payee identified by measuring their GPS locations, because the GPS location can be easily forged and the accuracy of the GPS sensor on the mobile phone is low. GPS signals are also blockable and spoofable. Furthermore, procedures that can improve the security of such measurement often require additional hardware support and complicated measurement processes. We can use HISPs to provide the security guarantee which can simplify the metrics of context to be captured and measured.

1.5.3 Evaluation of payment risks and solutions

We investigate possible attacks against electronic payments. We mainly focus on discussing credential harvesting, man-in-the-middle and man-in-the-shop attacks. Here, "man-in-the-shop" refers to cases where the merchant (or its staff) is not reliable or trustworthy. We summarise these risks and establish the attack model which is used to evaluate different payment solutions.

We investigate existing online payment solutions and mobile payment solutions to identify their advantages and disadvantages. In this thesis, online payments mean the customer pays the merchant online using a bank or credit card account; mobile payments mean that the payment is initiated and made on a mobile device such as a mobile phone.

We present an extensive comparison between different payment solutions and possible improvements. Our evaluation provides an overview of online and mobile payments, from which we develop the requirements of designing our new payment solutions.

1.5.4 Implementation of new payment solutions

We present concrete implementations which demonstrate details of how to use HISPs in practice ([17] shows some related videos). There are two main types of implementation:

1. Secure online payment. We propose the use of a trusted device to separate user account details and user inputs from the PC. The trusted device is a card reader with USB connectivity to a PC. It is implemented on a cheap 8-bit microprocessor with only 2Kb of memory. This device is connected to a PC, which is only responsible for routing data between the device and the merchant

or the bank. This device provides a strong protection against phishing, malware and man-in-the-middle attacks. It also allows the use of an electronic cheque which prevents man-in-the-shop attacks.

2. Secure mobile payment. We discuss two cases: (A) peer-to-peer mobile payment which supports direct payment between two mobile devices; (B) customer-to-merchant mobile payment which supports payment between the customer and the online merchant. Our proposal for a unified mobile payment platform allows us to build new mobile payment applications that can support all electronic connections, including online payment.

1.6 List of publications

1. Chen, B., Nguyen, L. and Roscoe, A.W. (2012) *Reverse authentication in financial transactions and identity management.* In Journal of Wireless Networks, Mobile Networks and Applications, 2012.

2. Chen, B. and Roscoe, A.W. (2012) *Social Networks for Importing and Exporting Security.* In Proceedings of the 17th Monterey Workshop on Development, Operation and Management of Large-Scale Complex IT Systems, 2012.

3. Chen, B., Flechais, I. Creese, S., Goldsmith, M. and Roscoe, A.W. (2012) *Secure Communication in Disasters.* In Proceedings of the Designing Interactive Secure Systems (DISS) Workshop, 2012.

4. Chen, B., Nguyen, L. and Roscoe, A.W. (2011) *When Context Is Better Than Identity: Authentication by Context Using Empirical Channels.* In Proceedings of 19th Security Protocols Workshop, 2011.

5. Chen, B. and Roscoe, A.W. (2011) *Mobile electronic identity: securing payment on mobile phones.* In Proceedings of the 5th IFIP WG 11.2 international conference on Information security theory and practice: security and privacy of mobile devices in wireless communication, 2011.

6. Chen, B., Roscoe, A.W., Kainda, R. and Nguyen, L.(2010) *The Missing Link: Human Interactive Security Protocols in Mobile Payment.* In Proceedings of the 5th International Workshop on Security, short paper, 2010.

7. Huang, X., Chen, B., Markham, A., Wang, Q., Zheng, Y. and Roscoe, A. W. (2012) *Human interactive secure key and ID exchange protocols in body sensor networks.* To appear in IET Information Security, Special Issue on Trust and Identity Management in Mobile and Internet Computing and Communications, 2013.

8. Huang, X., Ma, X., Chen, B., Markham, A., Wang, Q., Zheng, Y. and Roscoe, A. W. (2012) *Human Interactive Secure ID Management in Body Sensor Network.* To appear in Journal of Networks, 2012.

9. Huang, X., Wang, Q., Chen, B., Markham, A., Jäntti, R. and Roscoe, A. W. (2011) *Body sensor network key distribution using human interactive channels.* In Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies, 2011.

   My work is mainly to study how to use HISPs in payments and I am recently working in other applications. All the protocols used in this thesis are closely based on ones designed by Nguyen and Roscoe [119, 120, 121, 123, 122, 138]. Similarly the digest function used were developed by them [124]. I am working jointly with Xing Huang in studying how to use HISPs in body sensor networks.

1.7 Summary of thesis structure

Chapter 2 gives the background and related research. Chapter 3 reveals the attributes of payments where knowing whom to pay is the first security question to be solved before making any payments; we then present the method of reverse authentication which is the basis of all payment solutions developed by us. Chapter 4 and 5 evaluate the risks and existing solutions of online and mobile payments. We compare different payment solutions and establish the requirements for developing new payment solutions by using HISPs. We then implement our own payment solutions based on these requirements. Chapter 6 gives a proposal of developing a unified mobile payment platform on which various novel payment solutions can be built. Chapter 7 gives our conclusions and introduces further usages of HISPs and proposals for future work.

Chapter 2

Background and related research

2.1 Introduction

This chapter gives necessary background knowledge of payments. We also provide a general discussion of related research.

Traditional e-payments normally take one of the four different forms [139]:

1. e-cash or cash-like payments. The customer first creates a certain amount of e-cash by transferring from his/her bank account. The merchant receives e-cash from the customer and sends it to its own bank which will ask for a settlement from the appropriate authority.

2. Credit card and cheque payments. The customer fills in or creates an electronic form which contains the necessary information of his/her card account and the payment amount. The merchant receives this electronic form and forwards it to its bank to ask for a settlement from the customer's bank. Credit card payments are essentially the same as cheque payments. We may consider cards as tokens to facilitate the creation of such an electronic form.

3. Remittance. The customer sends a command to his/her bank to send a certain amount of money to the merchant's bank account.

4. Debit order. The merchant regularly sends requests to the customer's bank to make a payment to its account. This is common when the customer requests a regular service, such as a newspaper, gym membership or accommodation.

Note that in remittance the customer does not need to expose his/her account information directly to the merchant. This is a useful feature when there are malicious merchants.

The use of e-payments keeps evolving. Except for the four types of e-payments mentioned above, new payment methods have emerged; for example, electronic wallets and card payment on mobile phones. We will discuss these in our case studies in Chapter 4 and 5.

A few protocols have been introduced to be used to secure payments; for example, the Secure Electronic Transaction (SET) protocol, the Internet Keyed Payment Systems (iKP) protocol and the Secure Socket Layer (SSL) protocol.

This chapter gives an introduction to the background of our protocol as well as those used in payments. Related research is introduced in this chapter, but more details are arranged to be discussed according to their topics in different chapters.

2.2 SET

Traditional SSL does not protect customers from merchants: merchants can know the card details of their customers. The consortium of major credit card companies and software companies published SET with the aim of providing improved security for online payments [156].

By using SET, the customer can send an "electronic cheque" to the merchant. This cheque, which is the purchase's request message, contains a dual signature and an encrypted version of the customer's payment information. A simplified version of the purchase request message [41] is as follow:

$$C \longrightarrow M : PIDualSign, OIDualSign$$

$$\text{PIDualSign} = Sign_{priSK_C}(Hash(PIData), Hash(OIData)), \\ Crypt_{pubEK_P}(PIData, Hash(OIData))$$

$$\text{OIDualSign} = \text{OIData, Hash(PIData)}$$

$priSK_C$ is the customer's private key; PIData is the customer's card details and other secrets; OIData is the order information; $pubEK_P$ is the payment gateway's public key. A payment gateway is an authority which verifies and authorises the payment.

This design allows the merchant to verify the purchase request message without knowing the payment information: it computes (Hash(PIData), Hash(OIData)) and knows the customer's public key. The merchant has to forward the encrypted part of this message to the payment gateway to get verified:

$$M \longrightarrow P : Crypt_{pubEK_P}(Sign_{priSK_M}(Hash(OIData), PIDualSign))$$

The payment gateway verifies Hash(OIData) and PIData without knowing OIData. Thus, the design of dual signature allows the cooperation between the merchant and the payment gateway, while maintaining the necessary privacy between the customer and the merchant. $priSK_M$ is the private key of the merchant.

However, there are two implications within this design that may undermine the security and the usability of SET:

1. SET does not specify the initiation of OIData. It requires third parties to define how to generate and authenticate the OIData.

2. Both the customer and the merchant have to obtain and install a key certificate.

It is critical to ensure the authenticity and integrity of the order information. Therefore choosing a trustworthy third party to deliver the order information is an important requirement of using SET. SET can become more self-contained if it can clarify the initiation of the order information.

The requirement of obtaining and installing a key certificate can become both expensive and inconvenient. The high cost of implementing SET is one of the reasons why it has never been used in practice [31]. In addition, an attacker may forge certificates which can defeat the entire system. We will discuss the risks of using a PKI in Section 2.4.

2.3 iKP

iKP was developed by IBM in 1995 [42]. It provides protection for online card payments by using PKI. "i" indicates the number of parties with a key pair involved in the protocol, for example, 1KP means only the service provider (e.g. banks or payment companies) needs to have a key pair. The customer sends all the information (the payment information and the account information) encrypted using the public key to the service provider. 2KP means both the merchant and the service provider need to obtain a key pair. This resolves the non-repudiation issue of the merchant because the merchant's digital signature is used. 3KP means the customer, the merchant and the service provider need to obtain a key pair respectively. This resolves the non-repudiation issue of the customer and the merchant because the digital signatures of both parties are used.

iKP is designed to make payments only: the designers assume that the customer and the merchant have already agreed on the order and price. In other words, it does not authenticate the order information. This is a potential weakness in the case that the assumption does not stand.

3KP requires that the customer, the merchant and the service provider need to obtain a key pair. We have discussed in the previous section that this can be expensive and inconvenient.

In the next section, we focus on discussing the risks of using PKI which is currently the most widely used security infrastructure in payments.

2.4 PKI

Public Key Infrastructures (PKIs) are commonly used to provide long-term security over the Internet. A PKI is based on the public key cryptography which uses a key pair to perform encryption and decryption. Such a key pair includes a public key and its corresponding private key. The public key is published so others can see and use; the private key is kept secret by its owner to which the key pair is bound. A message

encrypted by a public can only be decrypted by its corresponding secret key and visa versa. Encrypting a message using a secret key is called signing.

A public key is distributed to the public in the form of a certificate which includes the public key, the key owner's identity information(e.g. a name, a domain name, or an email address), the key issuer's details, and other necessary information of using that public key. A public key certificate is signed by a Certificate Authority (CA) which is responsible for authenticating the binding relationship between the public key and its corresponding owner.

Before using a certificate, the user will first check its issuer's digital signature. This is often achieved by installing a set of trusted Root Certificates in the web-browser. This means the trust is achieved in two parts: (i) the user must trust the certificate issuer or the CA; (ii) the user must trust the web-browser or the developer of the web-browser. We organise our discussion accordingly.

2.4.1 Certificate Authority

The number of Internet CAs keeps increasing, with presently more than 600 [5] located in 52 different countries [62]. This raises a number of security concerns, but most importantly undermines the security model underpinning the PKI, as it is impossible to ensure that all of these CAs are responsible and secure. A number of incidents relating to the compromise of CAs have been reported in 2011; perhaps the most famous being that of compromised Dutch CA DigiNotar [2].

The consequences of compromising a CA can be serious. The security of a PKI is built based on the assumption that web-browsers, systems or applications can check the authenticity of a certificate issued for a certain organisation. However, if an attacker successfully breaks into one of these CAs, they can create fake certificates which are indistinguishable from genuine ones: they will appear to be issued by legitimate CAs. Therefore, if one of these CAs is unknowingly compromised, the security of the entire PKI may fail. This is a typical weakest-link problem. [125] introduces an example of improving the interoperability of PKI by introducing a verifying authority. Other approaches to improving PKI interoperability can be found in [72, 33].

2.4.2 Using certificate

We can inspect and manage the set of Root Certificates in our web-browsers. For example, on Firefox we can change the level of trust of a certificate according to the following three criteria: (A) trust this certificate for identifying websites; (B) trust this certificate for identifying email users; (C) trust this certificate for identifying software makers. A problem is that an ordinary user may not be able to determine what trust/right should give to a certificate given there are usually many different Root Certificates installed in a web-browser.

In addition, since the Root Certificates are installed by the web-browser provider,

it is difficult to determine which company is more responsible when installing certificates given there are many different web-browser providers. For example, on Google Chrome, a certificate issued under the name of MD5 Collisions Inc. has been identified as untrustworthy, while on Firefox, this certificate has not been included. The difference is that on Google Chrome the web-browser can immediately tell a certificate from MD5 Collisions Inc. is not trustworthy, while on Firefox the web-browser will ask whether the user trust this certificate or not.

The usability of PKIs is typically poor [63, 31], and users are regularly expected to verify the contents of a certificate. This presents a number of problems: firstly, users rarely check the details of a certificate, such as the issuer's name or the organisation's name it is bound to. Secondly, users can ignore warnings of unknown or fake certificates. Thirdly, the name contained in a legitimate certificate does not give any context to the entity it is associated with: (A) the name may be associated with a different entity with the same name; (B) even if the user checks the certificate, the name of an organisation contained in the certificate does not give any information about the rights and functions of the organisation: the trust of an authentic certificate does not directly infer trust.

PKI is only useful when participants already know one anothers profiles. Situations like inter-organisational cooperation, meetings or conversations require flexible, on-demand security: something which PKI solutions are too cumbersome to easily provide.

2.5 Related research in payments

This section introduces some related research on payments. More details will be presented in Chapter 4–6.

2.5.1 Online payment

The creation of online payments has been dominated by banks and credit card companies; for example, the well-known 3-D secure protocol (Verified by Visa and MasterCard SecureCode) and the Card Authentication Program (CAP) card reader. However, the authors of [61, 115] pointed out that these systems have serious security flaws.

There are solutions of using phone factors to enhance the security in online banking; for example, Authentify is a company providing online banking services by using telephony; Société Générale is a French bank providing a different version of the 3-D secure protocol which uses telephony to transfer the authentication code from the bank to the customer. However, these are subject to malware attacks and depending on using telephony to provide security may make it subject to attacks against telephony.

The authors of [87, 86] have pointed out that it is difficult to achieve strong security by using PCs for electronic banking. They suggest the use of a dedicated payment

device, for example, a trusted device, to make payment at home. The trusted device is often used to read bank cards and it has a USB connection to connect to the PC. In comparison to the CAP card reader, the trusted device can display information downloaded from the merchant or the bank, and then it can send back encrypted information which can only be read by the bank. This can eliminate malware attacks on the PC and significantly reduce the human efforts of inputting a large amount of information. An extensive discussion of potential threats against Internet Banking can be found in [86].

2.5.2 Mobile payment

Recent development on payment solutions focuses on using mobile phones as the electronic platform. This is largely because of their increasing pervasiveness and functionality. Some mobile payment solutions are similar to those we have seen on PCs; for example, most banks provide mobile interfaces to mobile web-browsers; some provide banking services via SMSs; and some provide dedicated mobile banking applications which provide similar or less functions functions we meet on PCs.

The majority of mobile payment solutions are used for peer-to-peer or customer-to-merchant payments. For example, the Paypal mobile application provides a service which allows users to make payment by bumping phones together; and NFC mobile payment solutions allows the customer to make payments to shops by simply touching their mobile devices to a touch pad. Innovations are frequently seen in this area; for example, mobile card readers are produced and released to allow customers to make card payments on their mobile devices. This quickly extends traditional card payments to mobile payments.

Some good examples of research in securing mobile payment can be found in [127, 103, 160, 128, 36, 29, 77, 35, 140, 67]. However, most research focuses on relying a trusted third party to manage and bootstrap security between the payer and the payee. This may not be convenient to implement in dynamic mobile payment environments. The authors of [57] presents a good literature review of previous research[1] on mobile payments. They also lists a few good research questions. We will discuss some of them in Chapter 6.

2.6 Conclusion

Traditional payment security has its limits which may lead to vulnerabilities especially when the payment environment is dynamic and ad-hoc. Since there are many different payment solutions, case studies are necessary to reveal urgent problems that need solutions. These are presented in Chapter 4 and 5. Before we move into the details of different payment solutions, we will present and discuss a security method which we will use to secure payment in the next chapter.

[1]The mobile payment research investigated in [57] is mostly made before 2006.

Chapter 3

Reverse authentication: using HISPs in payment

3.1 Introduction

This chapter presents a security method called the reverse authentication. It helps manage the payment relationship established between the payer and the payee, and it addresses the problem of identity management in situations where identity is not available. This chapter is mainly based on [50, 49].

Payments normally involve human actions[1], such as traditional methods like cash payment, and electronic payments like online payments and mobile payments. This provides a strong incentive to raise the following question: can we use HISPs in payments? In other words, can we make use of the human factor to help bootstrap the security that electronic payment requires? To answer this question, we first analyse cash payments.

Cash payments are probably the most popular and well-known payment method to people. We observe the following five properties of cash payment:

1. We clearly know whom to pay;
2. We clearly know the money has been handed to the intended payee;
3. We clearly know the amount of payment;
4. We lose at most the money being handed over;
5. Anonymity: cash payment does not require either party to know the other's name.

The first item means that, even though we may not know any personal information about the payee, we still clearly know he, she or it is the intended recipient. The second item means we clearly know the money has been delivered to the payee.

[1] In this thesis, we only discuss payments initiated by humans.

The third item means that we know exactly what we have paid. The fourth item means that, when there are risks we lose at most the money being handed out, but there is no risk of losing our personal information or account details. We believe that future mobile payment should have the above properties, and (depending on the implementation) anonymous payment should be a technological possibility. The anonymity property might well be desirable in some forms of mobile payment, since a customer might not want his or her identity to be recorded to a merchant, even though in most circumstances other than electronic cash they have to prove their identity to the organisation holding their funds.

We can see that the first security question to solve during any payment is knowing whom to pay. To answer this question, we will first analyse how we carry out authentications in our daily life in the following section.

3.2 Context vs identity

Context arises in an application when one or more entities are acting in a certain situation. For example, one of the most significant types of context is location, which can influence a wide range of decisions about whom to connect to across many applications.

We notice that context serves better than identity in some cases. For example, a customer C wants to pay a shop S. In this scenario, C knows he is in *this* shop and wants to pay it, even though he does not know its identity in a conventional sense.

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 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).

Therefore, we conclude that when it is difficult for ordinary users to correctly verify the identity of whom they are paying, context may be better than identity to help users to authenticate the payment. The difficulties of users may consist of two parts:

A. Users lack the necessary knowledge to correctly verify identity;

B. Users can be complacent, especially when the amount of payment is small.

We investigate how the payer (human) can authenticate that his device (typically a mobile phone) is connected to the intended payee. This authentication provides both assurance directly to the human and opportunities for improved transaction security such as better authentication of the human's identity.

We assume that there is a low-bandwidth empirical channel from the payee to the human that is not fakeable. This enables the human to play his part in HISPs. This is straightforward when the two are in the same place, and various ad hoc solutions

also work in remote contexts, such as e-commerce. There is more discussion about defining proper contexts in the next section.

We note that the payee and the payer usually both require authentication of the other. The payer needs typically to know he is paying the right entity for the transaction he is trying to complete. The payee, or more accurately, the infrastructure supporting payment (such as the banking system) needs to know that the payer is who he claims to be and that he is entitled to make the payment. As we have already noted, the first of these is frequently best attached to context. The latter on the other hand, is a much more typical process, and we note that much technology has been developed for this. It is also worth noting that, in most cases, the payee itself does not need to have the payer's identity information; rather, the assurance of its bank that it will make (or has already made) the payment. Therefore, the payee, in authenticating the payer, is acting as a proxy for the bank. Current technology typically gives the payee all information such as the credit card number, password/PIN, or at least makes it easy for this to be obtained. *This is undesirable as it offers the opportunity for abuses.*

Similar situations can be found in access control examples: in order to pass a check-point CP of building B, user C must submit his credentials (stored on his own device) to CP. In order to protect C's credentials, C needs to authenticate CP. In other words, he needs to know that his device is giving information to precisely this CP. But C does not know the identity of the person who stands in front of CP or there is no personnel at all. Therefore, only context that C can draw from the situation can help authenticate CP; for example, the location of CP (given that there is no other check-points standing there), the logo of CP, C's recognition of CP based on previous experiences, or somebody else C trusts tells C that this is the correct CP. In this case, there is only context rather than actual identity.

Another example can be found in social networks. Social networks are constantly changing our social styles and habits, and they are often considered our virtual presences on the Internet. Although different people may have the same name, the photos they share, the activities they join, the friends they have, and the profiles they present, provide a sophisticated body of context which can allow people to authenticate "who's who".

We can further observe that when context is better than identity, authentication by context brings more security than authentication by identity. To fulfil this requirement, we need first to obtain all the required contexts which would introduce the following three challenges:

1. It is difficult to define what proper context is or what is context after all.
2. Quite a few contexts cannot be automatically sensed by machines.
3. Some contexts can be easily forged.

The objective is to clarify the sort of situation in which context-based authentication is most appropriate.

3.3 Defining proper context

We borrow the definition of context from [58], thinking this best describes the meaning of context in the above cases:

> Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves.

However, this definition does not outline what is the proper context that can be used in a specific application, because the scenarios of applications differ in both the nature of applications and the environments of applications.

While context-aware applications are now widely available, we notice that some contexts, such as, logos, images, biometrics like faces and voices, cannot be easily recognized by machines (they may need special hardware support as well as a large amount of computation). And some contexts just simply cannot be sensed by machines at all; human trust, for example.

This creates a gap between a system and a human in terms of contribution to the process of bootstrapping security. We look for a method of authentication that can capture the properties of the process when we try to establish our trust in daily activities as described in Section 3.2. Such an authentication may be required in two different sorts of situation: remote authentication and on-site authentication. On-site authentication is the most natural way of allowing humans to make their own judgements: humans can see, hear and touch. Remote authentication can also provide humans an interface to sense if there is a proper proxy; for example, a phone call with a known individual, a PC and a *https* session, or a known social network web-page protected by https[2].

Such a method should be resistant to attacks using fake contexts. On the contrary, for example, locations, which usually appear in the form of addresses or GPS data (longitude, latitude, altitude), can easily be forged by presenting the false data; names, logos and photos can easily be forged, too. In order to achieve more pervasiveness, we do not rely on hardware or infrastructure support to solve this problem; we instead assume that humans, with or without enough knowledge, can make a choice between genuine and false context on behalf of their own risks.

The authors of [93] give an example of using a telephone line to authenticate location. Given that a telephone is located in a place where the user is, a phone call can be made to send a challenge to the user, and the user will send the challenge together with other information back to the service provider via another electronic connection. Such practices are common in financial services, for example, banks or credit card companies make phone calls to verify suspicious card transactions, and we have to answer secret authentication questions to get authenticated. We are interested in noting that even though we do not know the caller's phone number, we are still

[2]Facebook released its *https* service on 26th Jan, 2011[136].

willing to give our secrets to the caller who claim to be the bank or the credit card company. The context of when and where we receive this phone call, or the fact that the caller knows our name and the transactions we made or are being made, persuades us into believing that this call is authentic, even though it is subject to attacks, for example, man-in-the-middle attacks, phishing attacks, or insider attacks.

The discussion of security of hardware and software is beyond the scope of this thesis. Our focus is that, in either remote or on-site authentication, humans are able to find or create an empirical channel which is resistant to man-in-the-middle attacks. The question of whether we trust a telephone line to provide a good empirical channel is clearly open to different answers depending on the nature of threat and risk.

Before introducing our solution, we first assume that in any application that has a human presence, humans are capable of evaluating their own risks based on the context and then making their own judgements. Such context can be sensed either by human recognition or sensors.

In this way, the problem of defining what is the proper context for each application, and the problem of sensing contexts can be generalised as the user's choice of whether or not to trust the data received from the empirical channel. By incorporating humans into the design of the protocol, we can avoid the difficulties of defining specific scenarios and it provides a homogeneous way of incorporating trust into our protocols. In addition, the process of human interaction can be regulated by the design of implementation in order to eliminate unexpected human errors.

3.4 Introducing a HISP

To achieve security that allows intuitive authentication described in the previous section, we use a HISP which can efficiently exploit human efforts in bootstrapping security. It is closely based on the SHCBK protocol in [120, 121]. Because of its intended application we call the two parties C (customer) and M (merchant). Before the protocol is run, these two parties have no knowledge (such as shared secrets or knowledge of keys) to help them achieve security, except, naturally, the ability to run the protocol itself.

In order to run the protocol, each party must create two values of sufficient strength to achieve their cryptographic goals:

- Each party creates a *hash*, or *digest* key: we call these hk_C and hk_M. These are needed to randomise the calculation of the final SAS and we assume these are in the range 160-511 bits[3].

- C creates a session key k. This would normally be in the range 120–160 bits, but it could be increased to 512 bits (input width of the basic hash function) without penalty.

[3]The upper bound takes account of the common hash block size of 512 and the extra initial bit inserted by the protocol.

- M either creates freshly, or re-uses, an asymmetric key pair (pk, sk). There is no need for the "public" key pk to be certified. The length of these keys will depend on the desired level of security[4], the amount of available computing power, and the cryptosystem in use.

The protocol also involves a standard cryptographic hash function which possesses 3 main properties [109]: collision resistance, 2nd preimage resistance and inversion resistance. It depends heavily on the following property of such a function:

> If a party A has knowledge of $hash(V)$ for some value V, then while A cannot constructively compute V other than with infinitesimal probability, it can always check in future whether a value alleged to be V actually is V.

In this state A is *committed* to V but *without knowledge* of V. In the first pair of steps of the protocol, C and M both commit each other without knowledge of all the values. The only one of the four parameters hk_C, hk_M, pk and k communicated openly is M's public key pk:

1. $C \longrightarrow M : \ hash(0 : hk_C), hash(k)$
2. $M \longrightarrow C : \ hash(1 : hk_M), pk$

When these messages have been received, both parties are committed to values of all four parameters, but each lacks some knowledge. Thanks to the communication channel being insecure, they have no reason to believe that they are committed to the *same* values of the parameters – but of course they hope they are! Importantly, no intruder can know all four of the original (as opposed to hashed) values as created by the appropriate one of A and B.

The tags $0 :$ and $1 :$ are added to the hash keys hk_C and hk_M to ensure that the contents of these hashes can be distinguished as coming from a customer or merchant. This avoids the intruder reflecting hk_C back to C as a supposed hk_M in a way that C would ultimately accept.

Even if the intruder has participated in the protocol and impersonated one or both of the parties, it does not know the complete set of parameter values to which either C or M is committed. This is because no-one except C knows its value of hk_C, and similarly for M and hk_M.

The protocol now proceeds:

3. $C \longrightarrow M : \ hk_C, Encrypt_{pk}(k)$
4. $M \longrightarrow C : \ hk_M$

[4]The key certainly needs to be strong enough so that there is no realistic chance of it being broken during the life of the session being established. Further strength is required to ensure that the contents of that session remain secret after it ends, or re-usability of the key pair.

The second part of Message 3 is to tell C the actual value of the session key, which is now checked against the hash. In fact this transmission can be delayed until after the SAS comparison if the computation of the public key encryption $Encrypt_{pk}(k)$ is time consuming on a low-powered customer device.

It is the open transmission of the keys hk_C and hk_M at this stage that represents the core of the protocol. Firstly, of course, the participants must check that these are the same values that were used in constructing Messages 1 and 2. If not, the run is abandoned. Secondly, they (and anyone else who has been listening in) can compute a value for

$$digest(hk_C \oplus hk_M, (pk, hash(k)))$$

where $\oplus$ is bit-wise exclusive or and (X, Y) is an ordered pair. The protocol completes successfully if C (or C and M) are convinced that their two versions of the value – the SAS of this protocol – are equal: in becoming convinced they must not use a channel which can be "spoofed" by an intruder. Typically one will read their value to the other, or C will read M's value directly and compare it with her own. Whichever knows that the two values are equal can conclude that the link is authenticated. Typically this is either C or both of them. It is this comparison that makes it a HISP.

In the following paragraphs we will demonstrate why this protocol is secure. However, such demonstration of protocol security are notoriously hard to prove complete. A better approach is to use a verification technique. We have been applied this in [137]. Readers can find alternative arguments in [120, 121].

Naturally, if there is no attacker in presence, C's and M's values will be equal. If, however, an attacker has imposed his own values onto 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 $hash(k)$, so we will concentrate on what happens if the intruder interferes with these.

What we are concerned with is the chance that the digests agree when these two values do not.

The digest function [120, 121] 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 the theoretically optimal value of 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 also.

The following is an argument for why this protocol is secure. The reader can find similar arguments in [120, 121].

The intruder can easily convince either or both of C and M that the other's hk is not the one it should be. The only result of this activity, if done, is that as far as can be determined effectively by an intruder, C's and M's views of hk^* are independent uniformly distributed random variables. This is because of the following property of the bit-wise XOR used to construct it:

> If X and Y are independent random variables and X is uniformly distributed, then $X \oplus Y$ is uniformly distributed whatever the distribution

of Y.

Honest parties C and M will have chosen hk_C and hk_M uniformly at random, and independently of the other values in the protocol[5].

This means that the chance of C's and M's digests being equal is $\leq \epsilon$ whether their views of pk and $hash(k)$ are the same or not. We can conclude that it is not a good idea for the intruder to get C and M to disagree about hk_C and hk_M. So henceforth we will assume that they agree on these values: necessarily the ones they picked for themselves.

At the point just before Message 3 is sent, C knows that only she knows the randomly chosen hk_C. Therefore (whatever value was picked by M for hk_M), she has effectively randomised the final digest in a way that no-one else has knowledge of. M knows the same about his injection of hk_M. Each can therefore reason separately that the chance of agreement between the two instances of $digest(hk_C \oplus hk_M, (pk, hash(k))$ if their values of pk and $hash(k)$ are different is less than ϵ.

The crucial point is that each of pk and $hash(k)$ were known to both parties before the function that would be applied to the pair $(pk, hash(k))$ was known to anyone: indeed so far as any party is concerned, the value $hk_C \oplus hk_M$ determines that function remains uniformly distributed over all possible values until C and M have revealed hk_C and hk_M. It is this that means that there is no point in the intruder doing any combinatorial search against this function: by the time it is known, it is impossible to change the minds of C and M about what to apply it to.

Given that C and M do agree on $(pk, hash(k))$ it is also true that they agree on k, thanks to the assumed properties of $hash()$. When C sent $Encrypt_{pk}(k)$ she did not know who could understand it, but when she knows that pk is the value that M sent her she then knows that no-one other than herself and M know k. Similarly C, on receiving $Encrypt_{pk}(k)$, does not know from whom it came from, but once he checks that the hash of the key received this way corresponds to the value $hash(k)$ he and M agree to, he knows that he and C agree to M and that no-one else knows this value.

The above describes the principles behind this protocol which will be at the core of the payment method in this thesis. There are a number of others that operate in ways that, from the perspective of the human users, are either very similar or identical [68, 97, 100, 154], and see [123] for an extensive survey. A number of them can be adapted for group use. SHCBK, from which ours is adapted, was in fact *originally* designed as a group protocol.

Since HISPs fundamentally require the participation of humans, it is vital that their implementations are designed to optimise people's experience in using them, that they are designed to avoid users complacently ignoring the requirements on them, and make it easy to compare a reasonably secure digest or SAS length quickly. Research of these human factors issues can be found in [90, 89].

[5]The fact that hk_C is independent of the value of hk_M it accepts depends on the tagging of these values by 0 and 1.

3.5 Comparing the digest value

In many cases such as a financial transaction over the Internet or with a vending machine, or someone seeking to prove his identity and gain access to some service via a machine, there will only be one human present to perform the check of the equality of the digests or similar value used by other HISPs.

For high integrity applications involving a potentially complacent human in this way, there is a strong argument for having the human transfer a value manually rather than simply check that two displayed values are equal. So in fact the device actually performs the comparison – between the one its customer C has copied from merchant M and the one it has computed itself. There are still some interesting variants possible on this. In the following, we will split the entity C, representing the combination of a human customer Alice and her trusted device TD, into these two parts.

CA Most obviously and probably easiest in many applications: Alice reads a value from M and types it into her own device TD.

CS As a variant on this: when TD says that the two values agree, Alice presses a button on M to signal this agreement.

MA Alice reads the value from her TD and types it into M. In this case it is virtually certain that M will signal to the human whether the two values agree. This information may or may not be passed explicitly by Alice on to TD. (MA) will be the case where it is not.

MS This is the case like (MA) but where this information is transmitted by Alice to TD.

The pairs CA and MA, and CS and MS are clearly mirror images of each other: the first letter tells us who does the comparison in a method, the second tells us whether the resulting authentication is asymmetric or symmetric. In CA and MA only a one-directional *empirical* channel is used between TD and M, in opposite directions. In CS and MS, both sides are assured of equality provided our human is trustworthy and reliable.

Note that in cases CA and MA one or the other device proceeds without *knowing* that the digests actually agreed. Nevertheless these two cases are potentially useful: for example CA can be used when everything is confidential and of value that passes through the transaction moves in the direction from TD to M.

The most obvious case of this is Alice using TD to pay M for some goods and services: M wants to get paid but does not care that much who pays him, whereas Alice only wants to pay the merchant to whom she has an obligation to pay.

Notice that this reasoning does not apply when the merchant is passing value to a customer. Imagine that the merchant wants to pass value to the customer who is standing at a particular till, or is on a particular phone call etc. If the protocol is run

in mode MA it gains the assurance that it is this person's device that is connecting to it, since only she was in a position to make the empirical communication. In practice, however, this situation might well require the customer to pass (e.g. ID) information to the merchant that she would not want to give to *anyone*, and it would be better for Alice not to have to learn a very different protocol for a relatively unusual case. Therefore a protocol giving a symmetric outcome would probably be used here; in particular CS, with a warning to Alice about the consequences of pushing the final button incorrectly.

One way of more-or-less ensuring that she does behave correctly and not push the button without knowing that the digests agree is to have both sides compute a bit from the protocol parameters, and use it to have M only tell Alice which of *two* buttons to press once it knows the digests/SASs agree. The essential thing here is to make the customer look at the device that has checked equality before confirming anything to the other one.

We might note that it will almost always be Alice who identifies that M is what she wants to connect TD to, and that the connection process itself gives neither party any proof of the other's identity. Therefore, at the end of the HISP run:

- Unless MA is used, Alice knows that C is connected to M, as identified by being at the other end of the empirical channel.
- Unless CA has been used, M "knows" that the party it is connected to is the one at the other end of its empirical channel.
- Both know they have a shared secret symmetric key with the other, which can be used to secure and authenticate communication between them in a subsequent session.

3.5.1 Trust against trustworthy

Trust is something committed from one to one another; for example, the payer places his/her trust in the payee. Trustworthy, meanwhile, is a character of an object; for example, one may say that the bank's web-site is trustworthy.

In payments, there are situations where the payee is not known to be trustworthy, but the payer still needs to make a payment to him/her. For example, Bob is travelling and he needs to buy some food, but there is only one small shop he can find; although this shop is not trustworthy, he still needs to make the payment. In this case, security technologies can only enable that the payment has been made to the correct instance of the small shop, but can not ensure that the small shop delivers the right goods to Bob correctly. (But of course he can delay payment until he has the goods.)

Recall the five properties of cash payment in 3.1. The fourth property is to ensure the maximum risk is to lose the money being handed out. Therefore, we will only discuss how to design a secure payment technology that can at least achieve this property. Determining the trustworthiness of the payee is beyond the scope of our research.

3.6 Case study: supporting a financial transaction

The connection above has very little in common with the way that most personal financial transactions are performed, at least those involving banks[6]. For the current methods for doing these, whether manual (e.g. typing details from a credit card into an *https* site) or electronic (e.g., logging in to Internet banking; using Chip-and-PIN terminals at point of sale), they are designed simply to authenticate the payer (Alice and/or her credit card) to the payee and/or bank. Typically also, traditional methods of payment give the payee a great deal of sensitive information about the payer: note that anyone who has been paid on-line with a particular card has the information he (or anyone to whom it is unwittingly compromised) needs to pay for goods with that same credit card; and that nothing proves to Alice that the Chip-and-PIN terminal in her hands will not clone her card and remember her PIN.

A HISP is used to allow Alice to connect her TD to the particular merchant with whom she has been shopping, whether in person, on-line, or on the telephone. Clearly this authentication is in the reverse direction to the usual sort, as described in the preceding paragraph. This explains the title of this chapter: it can be viewed as *reverse authentication*.

What we have allowed Alice to do is to create a secure electronic connection with the merchant that she wants to pay, and furthermore where she is assured that the connection has been made within the context of the *particular* transaction for which she is willing to pay.

This electronic connection will allow a great deal more information to pass between her device (card/TD) that is otherwise possible unless she puts her card into the hands of the payee, or at all on-line.

In summary, what the reverse authentication achieves is not to *replace* the usual ID check on Alice and her device, but to make it potentially more thorough, particularly on-line, easier for Alice (because in most cases she will have to do less), and to remove the danger of Alice's secrets getting compromised. We will demonstrate this below.

We believe that beginning a financial transaction with the HISP authenticating merchant to the customer makes sense in all of the following cases.

(A) Electronic cash: the customer has some device with her that contains value, and she wishes to transfer some of that value to the merchant. It might well be the case that she wishes to do so anonymously.

(B) Credit card or cheque: the customer wishes to give the merchant the right to take a sum of money from her account. Note that, although the mechanisms are rather different, both conventional credit card transactions and paper cheques have this logical effect.

(C) Electronic banking: the customer wishes to give her bank a direct instruction to pay the money into the merchant's account. The logical difference with (B) is

[6] As we will see, good old fashioned cash transactions resemble our connection.

that the bank must be involved directly, and the merchant never holds a token that is good for money.

This first dimension influences the way the payment proceeds after a secure link has been established, and how identity issues arise. (A) is different from the others because our customer will not have to prove her identity (Alice) to anyone, while in the other two it is desirable[7] that she (separately from her device) proves that she is entitled to use the account by the entry of a secret PIN or biometrics.

In each of these we imagine a variety of payment situations, which influence how the authenticated connection is made:

(i) The customer is sitting at a desk and shopping on-line. Here we assume that the customer and the banking system do not trust the PC except possibly through an *https* windows displayed on browsers. [This is not to say that this last mechanism is 100% secure, but since e-commerce and digital banking rely on it currently, it seems reasonable that we do not increase the risks inherent from using it in present methods[8].]

(ii) The customer is trying to pay the merchant in person: in present technology she would hand over cash or credit card, or place her card in a reader presented by the merchant. Here, the merchant might be a machine or be a manned till.

(iii) The customer is shopping over the phone.

In case (C) (mobile banking) we do not concern ourselves with how the secure connection between the phone and bank is made, since we can reasonably assume that there is a long-term key which achieves this. In all cases, we assume that a HISP is to be used to connect the phone to the merchant that is to be paid, thereby proving to Alice that she is paying the correct entity within the correct transaction.

Except in the case where the paying device TD is the same telephone over which the transaction is being conducted, we need to get some connection between TD and merchant established so that it can be authenticated with a HISP. In technology used for everyday payments, both of these phases need to be very easy. One advantage of using a HISP is that the customer's view of the second phase can be the same in every case. The following sets out a few options for making the initial insecure connection in the on-line and point-of-sale cases.

In the online case there are two main options for this connection: the first is using the home PC on which the shopping is being done as a link between TD and merchant. The link could then be made by wire (e.g. USB), wireless (e.g. WiFi or Bluetooth) or infrared. None of these is technologically difficult, and the session on the browser can instruct the PC about where to route the communications.

[7]We note that in some present credit card transactions, especially on-line ones, she does not have to do this.

[8]In fact we argue that our methods provide a higher degree of security than the traditional use of *https* sites, since we are only relying on the communication through it being authenticated, not secret. Thus neither screen-shot grabbing nor key-sniffing would benefit an attacker.

The second is using telephony to make the connection: this may be the only option when Alice is forbidden to connect any personal device to the PC. The only problem then is giving one or the other side of the connection (phone=TD or merchant) the information required to connect telephonically to the other. This will be the combination of a telephone number and a (probably one-time) token that identifies the particular transaction. The merchant's number can be transferred to the phone from the PC by (e.g.) Bluetooth, but of course this would fall foul of the no-connection rule if this applied. The user's number can be pre-loaded into the browser (and there are strong arguments for this being a separate number from the one used for ordinary phone functions), and this sent together with a one-time token to the merchant when a button on the payment site is pressed.

In the point-of-sale case, the same two options (local and telephonic) connection apply. A literally wired local connection is unlikely, but it would be possible to place a phone into a special cradle. In that case it is probably not necessary to use a HISP, since the phone is obviously connected to the merchant. Similarly, if a connection is bootstrapped from a physical connection that Alice can *see*, this is still probably not necessary, and the same may apply for low-value transactions if it is bootstrapped from very short range radio as used for example, in Oyster cards [9], provided this generates a session key. A HISP will provide additional assurance in this last case for high-value transactions. Other options include Bluetooth connection (where it may be necessary for Alice to select which till she is at) or telephony (where the number can be transferred using any of the methods described above, or perhaps via scanning of a bar-code displayed on the phone).

In all payment methods, we assume the first action after the secure session is established would be for the merchant to send the TD details of the transaction it wishes to be paid for plus secondary security information such as its name and logo. If Alice agrees to the payment, she will either press a button or enter the personal information (e.g. PIN or biometric) needed to confirm her presence. Whatever payment token is then sent by her TD will then contain the secondary security information so that a fake merchant who has "borrowed" these should not be able to obtain payment from them.

An electronic cash payment would simply follow the appropriate protocol over the secured session.

For a credit card transaction we either have greatly increased the communication possibilities between TD and merchant or (particularly when Chip-and-PIN terminals are replaced) ensured that there is much less availability of customer information to merchant. In either of these cases it makes sense to replace present payment methods by the TD giving the merchant an e-cheque containing

- Payee, payer, amount, credit card details and time-stamp, as on a conventional cheque.

- Transaction ID.

- Any secondary security information about the payee that Alice has confirmed. *The bank will confirm that this ties up with the payee.*
- Evidence that Alice has correctly proved her own identity. This might be either the actual information (e.g. PIN) she has input, or evidence both that the TD/card has confirmed this information (noting that, at present, PINs are typically confirmed by a credit card and not transmitted) and that the TD/card itself is genuine and behaving properly.

This would be encrypted under a key that merchants cannot understand (e.g. a symmetric key specific to this TD/Card or the public key of the banking system) and sent to the merchant to be forwarded and authorised by the banking system.

Perhaps the most attractive scenarios for using HISPs for payment comes in the context of *mobile banking* (i.e. on-line banking on a mobile phone). Systems implementing this with limited functionality are rapidly being developed by banks and rolled out to customers, but none that we are aware of allow the user to make a payment at a general point-of-sale or on-line merchant. The deficiency can be remedied once the phone is securely connected by HISP to the merchant. For then M sends details of the transaction for Alice to confirm, plus bank account details to which the money is to be paid. When C confirms and gives whatever authorisation code is required by her bank, the on-line banking session automatically generates a transfer to M's bank, and an unforgeable certificate that this has occurred is sent by C's bank to M via C. The value of the HISP here is that it ensures that the bank account details really come from M.

It is worth noting that the total effort that Alice has to make in running a HISP and confirming the transaction on her TD is substantially less than that is required of her in conventional on-line purchases using credit cards, whether these are performed by entering card details onto a web-site or by entering her PIN into a secondary device provided by the card issuer and then copying a one-time authentication code into the web-site. (Devices such as these are, of course designed to help Alice prove her identity – they do not help Alice to create an authenticated connection to the merchant.)

3.7 You, me, us and anonymous authentication

One of the strangest qualities that HISPs have is that they provide essentially *anonymous authentication*: they can provide security in contexts where the parties being connected do not know each others' names. We can imagine this being extremely useful in some applications such as electronic cash where a payee and payer might want to be assured that it really is them who are connected for some transaction without actually revealing their names to each other. The curious thing is that this situation is completely familiar when carrying out transactions with good, old fashioned, cash. What HISPs can do is to enable human(s) to connect devices that they trust because of their context without knowing their names.

In many situations the personal pronoun *you*, whether singular or plural, typically indicates that the speaker is addressing a collection of people that both the speaker and they understand precisely (i.e., what collection of people is being addressed), without carrying any implication that the participants know each other's names. The sort of authentication given to a single individual by running a HISP with such a collection might thus be termed *you*-authentication, while the sort achieved by a group who are all aware of the final agreement might be termed *us*-authentication for similar reasons.

Thus these protocols allow humans to build networks of secure communications amongst the devices that they trust, reflecting the same intuitions that guide their everyday life. For example, each of us develops a good sense of when to hand over money, or our credit card, to pay for goods and services, and who to hand these things to. This sense is largely unrelated to our knowing the identity of the person or other entity we are paying. Indeed, in many cases, one either does not know this identity – how many times do you look carefully at the name of the filling station where you buy petrol? –or this name is largely irrelevant – how would it change your behaviour if you had noticed this name? What matters is context: the payer believes that the payee either has or will supply whatever it is that she is paying for, and accepts the contract to supply these in return for money. What our protocols achieve in this context is that they allow the payer to connect her account to the payee over an insecure network, so that the link becomes secure and she knows that the communication partner is the same entity that her intuition would have told her to pay by more conventional means.

Similar situations will arise whenever a party Alice carries some piece of paper or a card that either carries some less-than-public information (often about her), or which enables something of value to be obtained or transferred. Examples are a passport, driving licence, ID card, medical details (whether on a record card or derived from some sensor attached to Alice), company pass, tickets for travel, and membership cards. If these things are held on some electronic device, then Alice will want to know they are connected to the entity that she trusts with this information (the same one to which she would be happy to hand over the same details on paper). HISPs appear to be the ideal vehicle for this.

The role of her handing over this information to Bob (analogous to M) may simply be to prove her identity to him. She may be very reluctant to let anyone else see this information because of the dangers of her identity being stolen.

It is interesting to reflect at this point that she may well be interested in proving her identity to Bob without giving him the wherewithal to steal it, or "mis-lay" her information by not storing it carefully enough.

There is a potential advantage to Alice in the "card" or similar remaining in her possession on her own electronic device rather then being handed over to the recipient. This is that she can limit the transfer of information or value to what are necessary for the particular transaction, whereas a physical card or piece of paper might well be copied either as part of the normal procedures of Bob, or if he were less trustworthy

than Alice believes. Even the first case is dangerous to Alice unless Bob protects her information better than has frequently been the case. Our electronic payment scenario illustrates this perfectly: if Alice hands Bob her credit card, she is giving him the means to charge unlimited transactions to her account. On the other hand, if she merely connects her account to him, she can hand him an e-cheque, usable one-time only for a stated amount of money. The latter would not be practical without an electronic connection, because it is too large a piece of data to transfer by hand.

Thus the methods we have devised for payments is clearly a useful model to consider for pure identity verification, with the financial component removed.

Just as in the case of a payment, the particular entity Bob with which Alice connects may only be *semi*-trusted by her. In the payment case, she wants to avoid disclosing long-term secrets to Bob and only give him the exact amount of money she is trading with him. In other cases she may only want to give the exact amount of information required by Bob's apparent function, and may wish (as in the payment) to delegate the confirmation of her identity to a third party whom she and a legitimate Bob *fully* trust. As in the e-cheque case, the third party may sit behind Bob, or, as in the e-banking case, it may sit behind Alice. Depending on the context, there may be also be analogues of the secondary security information used in payments.

3.8 Conclusion

The payment relationship we have observed today often ensures the authentication of the customer. As the development of electronic payment continues, we will see payments take place more frequently in more ad-hoc scenarios. In these cases, it is difficult to assume that the customer will have adequate knowledge of correctly knowing the details of the merchant. Our method of reverse authentication gives the customer the convenience as well as security to first authenticate a connection to the merchant using a HISP even if he/she does not know the merchant's name. The customer can then use this connection to securely complete the payment.

We believe that this technology will have many applications both within the area of financial transactions highlighted here and more widely. We will demonstrate its usages in securing online and mobile payments in Chapters 4 and 6.

Chapter 4

Online payments

4.1 Introduction

This chapter addresses the problem of securing online payments. In this thesis, we define online payments as electronic payments made in connection with an e-commerce transaction occurring with an *https* session between a browser and the merchant. We concentrate mainly on the case where the browser is running on a PC. In particular, we focus on evaluating web-based card payments, e-wallet payment services and online banking. We examine the benefits of technology that allows the customer to establish a secure connection which authenticates the merchant. We also build an evaluation model to analyse a few selected solutions.

There is a growing trend to replace paper cheques with secure electronic systems in the financial world. For example, the board of the UK Payments Council made an announcement on 16 December 2009 that cheques were to be phased out by 2018. However, this decision has been reversed since there are no mature technologies to enable it [11].

The recent development of authentication technologies in the financial world shows how banks are eager to protect themselves and their customers from fraud. They are moving from traditional one-factor authentication to two-factor or multi-factor authentication. However, many of them barely provide a way to help the customer to authenticate a payment. Instead, liabilities have been constantly pushed from the bank to the customer's hands accompanying the deployment of new authentication technologies [61, 115]. The new technology frequently involves the customer in a significant amount of effort, as well as leaving open some documented vulnerabilities.

In the following sections, we review some existing solutions to online payment security. We examine the complexity of these solutions for the customer, potential vulnerabilities and ways in which they could be improved by the prior use of a HISP. We will give an example HISP which could play this role, and measure its usability and security as well. An experimental implementation is made. We focus on discussing the viability of implementing a secure and inexpensive card reader with connection to a PC. Importantly, besides offering extra security to the customer, our

solution considerably reduces the effort involved and improves the authentication of the customer.

4.2 Evaluating risks

In this section we introduce a few existing risk evaluation methods from which we will select one to conduct our risk evaluation. Risk evaluation is to identify threats (e.g. what are the potential threats of a certain asset?), determine vulnerabilities to threats (e.g. how vulnerable of a certain asset is to a certain threat?), estimate risks (e.g. how big is the potential loss of a certain vulnerability being exercised?), and find out directions for risk mitigation solutions. Our risk evaluation is based on the data gathered from interviews with managers and experts, documents provided by payment companies, literature review of academic papers, online vulnerability databases, and tests of existing payment solutions.

In general, there are two types of risk evaluation methods: quantitative methods and qualitative methods. Quantitative risk evaluation is mainly based on calculating the potential financial loss of a threat becoming a reality in terms of "dollars". Good examples are Annualised Loss Expectancy (ALE) [134] and Courtney [55]. Table 4.1 shows their formulas of calculating the total IT risk exposure.

Table 4.1: Formulas of quantitative risk evaluation methods.

Method	**Formula**	**Notes**
ALE	Risk $= \sum_{i=1}^{n}(V_i \times EL_i)$	V_i = vulnerability = probability of occurrence per year. EL_i = expected loss = expected loss of ith threat/vulnerability pair.
Courtney	Risk $= \frac{10^{(p+v-3)}}{3}$	p = an integer representing orders of magnitude of estimated frequencies of a loss. v = an integer representing orders of magnitude of the dollar impact of an asset's loss.

The result of quantitative risk evaluation methods is the loss of dollars which is easy to understand and compare. However, using these methods requires a good understanding of the values of assets and the value of the potential loss of a threat to an asset, and the frequencies of the loss. This is difficult to achieve by us since we do not have the necessary data to make accurate estimations of (A) values of assets and hence values of losses and (B) frequencies of losses.

One alternative is to adopt qualitative risk evaluation methods. Qualitative risk evaluation methods express risks in terms of descriptive variables. There are many qualitative risk evaluation methods. Some early examples are documented in [134]. [134] introduces three kinds of qualitative methods:

1. Scenario Analysis. In this method a group of experts list various scenarios where assets might be subject to loss from threats. Then they rank different scenarios according to their importance. This method can quickly identify the weakest parts in a system.

2. Fuzzy Metrics. This method labels fuzzy descriptors to different metrics. For example, to evaluate the probability of loss, we may have values of low, medium and high. Further we may assign numerical values to fuzzy descriptors in order to compute risks. For example, we may define values of probability as: low = 0.1, medium = 0.5 and high = 1.0.

3. Questionnaires. This method tries to obtain opinions and statistics from computer vendors, security companies and publications on computer security.

According to the descriptions of the above methods, we understand that the data we have collected is suitable to be used in a qualitative risk evaluation.

Similar methodologies are found in more recent examples. We have selected and investigated risk evaluation methods from three organisations:

1. National Institute of Standard and Technology (**NIST**). We have investigated two sources (from NIST) related to risk evaluation: (i) "Special Publication 800-30: Risk Management Guide for Information Technology Systems" (SP 800-30) [148] provides guidance on risk evaluation and management. When computing risks, it uses two metrics: Likelihood and Impact (when determining Impact levels, it suggests using three sub-metrics: Loss of Integrity, Loss of Availability and Loss of Confidentiality). (ii) Common Vulnerability Scoring System[1] (CVSS) is an online scoring system which can compute the score of a vulnerability by assigning values to various metrics (in CVSS Version 2.0 there are 3 metric groups and 14 metrics in total).

2. **Microsoft**. Microsoft provides a powerful tool called Security Development Lifecycle (SDL) to help identify and manage risks. It allows the developer to draw a diagram of the data flow of the system and then it automatically produces a report of potential attacks and countermeasures. Before the release of SDL, Microsoft published guidance for identifying attacks and rating risks [108]. When computing the risk of an attack, it uses a model called DREAD. DREAD represents five quantifiable metrics: Damage, Reproducibility, Exploitability, Affected users and Discoverability.

[1] `http://nvd.nist.gov/cvss.cfm`

3. Carnegie Mellon University's Software Engineering Institute (**SEI**). Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE[2]) is a comprehensive and sophisticated risk evaluation and management methodology developed by SEI. It is designed for organisational security analysis and provides strategical guidance on managing security. In their documents, they use a medical information system as an example. When computing risk levels, OCTAVE assumes the probability of an attack is always "1", in other words, it assumes that if there is a vulnerability, the attack of exploiting this vulnerability will always happen; and it uses six metrics to determine the impact level of an attack: Reputation/Customer Confidence, Life/Health of customers, Productivity, Fines/Legal Penalties, Finances and Other (Facilities).

Our goal is to find a "lightweight" method of computing the risk level of an attack. We must simplify our method of analysis in order to evaluate different cases. For example, we plan to cover multiple cases in this and next chapter; and each case will be presented in a single section. To achieve this, we only focus on the critical part of the payment process instead of the entire payment system; and we only evaluate the most significant attacks instead of all potential attacks.

Existing risk evaluation methods often employ sophisticated processes to identify potential risks, but they are not particularly designed for payment systems. For example, OCTAVE (18 volumes in total) requires a concrete description of the entire system and efforts of working with a range of worksheets and practices, and it focuses on organisational risks rather than technical risks; SDL requires the user to draw a diagram of the data flow of the system and it intends to expose all possible threats/vulnerabilities in the diagram.

When choosing a method for computing risks, we discover that some methods are not suitable for payment systems. For example, OCTAVE requires measurements of Life/Health of customers, which are not only difficult to estimate but also unnecessary when evaluating payment systems. A straightforward criterion is to choose one method with the smallest number of metrics to measure. Table 4.2 shows the number of metrics used in different methods of computing risks.

SP 800-30	CVSS Version 2.0	DREAD	OCTAVE
4	14	5	6

Table 4.2: The number of metrics used in different methods of computing risks.

We adopt the risk computation method of SP 800-30 because: (i) it requires less metrics to measure according to Table 4.2, which helps us to present our case study in a more compact format; (ii) it provides a detailed introduction of risk assessment including system characterisation, threat identification, vulnerability identification,

[2] `http://www.cert.org/octave/`

control analysis, likelihood determination, impact analysis, risk determination, control recommendations and results documentation. In addition, according to [43] SP 800-30 is less aggressive in giving high risk values to attacks compared to DREAD and CVSS. This helps us to reduce the number of prominent attacks so that our discussion can become more focused. However, since SP 800-30 is not designed particularly for payment systems, we need to make necessary modifications to certain rules and standards in order to use it in our analysis.

4.3 Introducing SP 800-30

To make our discussion consistent, we adopt the following concepts defined in SP 800-30:

> **Risk** is a function of the likelihood of a given threat-source's exercising a particular potential vulnerability, and the resulting impact of the adverse event on the organisation.
>
> **Threat** is the potential for a threat-source to exercise (accidentally trigger or intentionally exploit) a specific vulnerability.
>
> **Threat-source** is either (1) intent and method targeted at the intentional exploitation of a vulnerability or (2) a situation and method that may accidentally trigger a vulnerability.
>
> **Vulnerability** is a flaw or weakness in system security procedures, design, implementation, or internal controls that could be exercised (accidentally triggered or intentionally exploited) and result in a security breach or a violation of the system's security policy.

Note that in order to simplify our discussion, we only focus on human instigated threats. We will not discuss natural threats (e.g. floods, pollution, storms, earthquakes) in this chapter.

Figure 4.1 shows the threat analysis process of SP 800-30. Step 1 is to define the boundaries of the IT system and identify assets to be evaluated. Step 2 is to identify threat-sources and their motivations and threat actions. Step 3 is to identify and list vulnerabilities associated with the system environment. Step 4 is to identify existing controls (or planned controls) and determine their effectiveness. Step 5 is to analyse the probability of a threat-source exploiting a vulnerability. Step 6 is to determine the impact level resulting from a successful threat exercise of a vulnerability. Step 7 is to calculate the risk level by multiplying the likelihood of a vulnerability being exercised with the magnitude of the impact resulting from such an exercise of the vulnerability. Table 4.3 shows an example risk-level matrix documented in SP 800-30. In our analysis we adopt the scales and values used in Table 4.3. Step 8 is to give control recommendations. Step 9 is to present evaluation results in a report.

We organise our evaluation according to the above 9 steps. Table 4.4 shows the arrangement of different sections.

Threat likelihood	Impact level		
	Low (10)	Medium (50)	High (100)
High (1.0)	Low $10 \times 1.0 = 10$	Medium $50 \times 1.0 = 50$	High $100 \times 1.0 = 100$
Medium (0.5)	Low $10 \times 0.5 = 5$	Medium $50 \times 0.5 = 25$	Medium $100 \times 0.5 = 50$
Low (0.1)	Low $10 \times 0.1 = 1$	Low $50 \times 0.1 = 5$	Low $100 \times 0.1 = 10$

Risk Scale: High (>50 to 100); Medium (> 10 to 50); Low (1 to 10)

Table 4.3: The example of risk-level matrix according to SP 800-30

Note that the values of scales used in Table 4.3 are subjective, hence the results of the computation are not accurate risk scores: they cannot be used without the current context. The purpose of quantifying risks is to show the relative performances of different payment solutions.

We clarify a few terms before presenting our risk analysis. We use the word "attack" to describe a threat action. This is to facilitate our discussion since "attack" is more convenient to describe the action of a threat-source exercising a vulnerability. We use the word "attacker" to replace the word "threat-source" when necessary since we only focus on human instigated threats.

Steps in SP 800-30	Sections
Step 1. System characterisation	Section 4.4
Step 2. Threat Identification	Section 4.5
Step 3. Vulnerability Identification	Section 4.5
Step 4. Control Analysis	Section 4.5
Step 5. Likelihood Determination	Section 4.6
Step 6. Impact Analysis	Section 4.6
Step 7. Risk Determination	Section 4.7
Step 8. Control Recommendations	Section 4.7, 4.8
Step 9. Results Documentation	Section 4.4 – 4.8

Table 4.4: The organisation of sections.

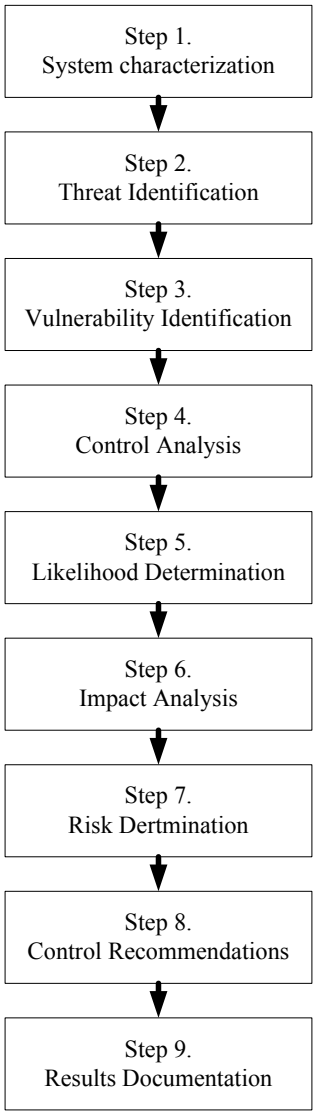


Figure 4.1: SP 800-30 risk evaluation process.

4.4 System characterisation

Figure 4.2 on page 45 shows an example of the whole system. In general a payment system involved in one payment may include up to four classes of agents:

1. A private individual with his own hardware. For example, when making a payment at home, one may use his own PC to make payment. In practice, more hardware may be involved. For example, cards and card readers. These are described as the customer payment system which directly interacts with the customer.

2. A merchant with its own server or PC. It receives payment from an individual. An individual or a company can play the role of merchant. These are described as the merchant payment system. It is responsible of managing card details/money/cheque sent from the customer, verifying and completing the payment, and sending back receipt to the customer.

3. An e-wallet provider with its server. The e-wallet provider is sometimes called the third-party payment company. This differentiates its role from the bank. It provides online payment accounts to customers. Note that in practice, if the merchant receives payment by using an e-wallet account, the customer needs to use an account from the same e-wallet provider. For example, this is true when the merchant receives payment using a Paypal account. Therefore, we assume there is only one e-wallet provider involved in a particular payment.

4. One or two banks. For example, the customer and the merchant may use bank accounts from the same bank or from different banks. We therefore overlay two banking systems in Figure 4.2. The number of banks involved depends on context.

Connections A, B, C, D, E, F and G represents possible electronic communication between different agents. Figure 4.2 helps to describe different ways of making payment. We list these as follows:

1. Payment by using e-wallet accounts. Money is transferred from the customer's e-wallet account to the merchant's e-wallet account. Route (A, B, C) represents this example: the customer interacts with the merchant via connection A to make the purchase, then the customer sends payment command to the e-wallet provider via connection B; after the money is transferred from the customer's account to the merchant's account, the e-wallet provider informs the merchant via connection C.

2. Payment by using bank accounts. Money is transferred from the customer's bank account to the merchant's bank account. Route (A, D, E) represents this example: the customer interacts with the merchant via connection A to make the purchase, then the customer sends payment command to the bank via connection D; after the money is transferred from the customer's account to the merchant's account, the bank informs the merchant via connection E. When there are two banks involved, Route (A, D, E) becomes Route (A, D, G, E).

3. Payment from the customer's bank account to the merchant's e-wallet account. In practice, the customer still needs to register with an e-wallet provider in order to use this payment service. For example, you can pay by using your bank account to another Paypal account via Paypal. Route (A, B, F, C) represents this example.

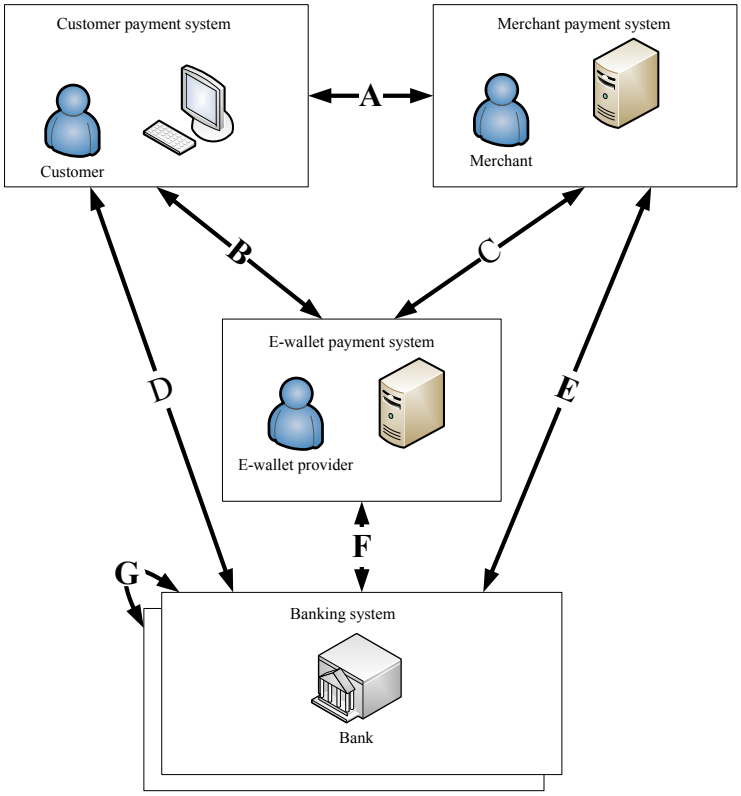


Figure 4.2: An example of the payment system structure.

4.5 Establishing the attack model

This section discusses Step 2 – 4 in SP 800-30. Based on our analysis we will generate an attack model which is a list of potential attacks against online payments.

SP 800-30 suggested five types of human threat-sources: hacker, computer criminal[3], terrorist, industrial espionage and insider. In payments we have identified two threat-sources:

1. Computer criminal. The motivation for a computer criminal is monetary gain. We assume that a computer criminal has control[4] over the user's PC. This is achieved by installing malware on the PC. We assume that a computer criminal has control over the communication between the user's PC and the merchant.

[3]According to SP 800-30 the difference between a hacker and a computer criminal is that a computer criminal is motivated by monetary gain while a hacker is not. In payments, we may simply assume that the motivation behind all threats is monetary gain.

[4]This is not absolute, for example, the user can have limited trust in the browser in some applications.

For example, he/she can eavesdrop, intercept and modify data sent by a customer or a merchant. A computer criminal is capable of exploiting vulnerabilities exposed in the entire payment system.

2. Insider. The motivation for an insider is also monetary gain. We define an insider as a malicious person who has access to the resources in the merchant payment system or the e-wallet payment system. This definition is broader than others where insiders are existing employees or former employees. He/she may modify or steal data, tamper with instruments, or sabotage system infrastructure. An insider is capable of exploiting vulnerabilities exposed in the merchant payment system and the e-wallet payment system.

Note that we do not examine threats within the banking system since it is difficult to obtain necessary information of the entire structure of the banking system. We therefore assume that the banking system is secure to reduce the range of our discussion.

Three vulnerability databases are used in our research: the National Vulnerability Database[5] of NIST, the Vulnerability Notes Database[6] of United States Computer Emergency Readiness Team (US-CERT), and the malware database[7] of Kaspersky Lab.

As well as technical vulnerabilities, we discover that complacent humans often cause security problems in payments. We therefore make the following assumption to generalise the human vulnerability:

> Humans may choose weak passwords, ignore security warnings, skip necessary security processes/actions when they are allowed to; and humans may make mistakes when verifying information (e.g. comparing names, numbers and other information).

Techniques for controlling threats vary from system to system. It is impractical to list and discuss all details of techniques for every example we choose to discuss in this chapter. Therefore, we generalise them into the following groups:

1. Anti-virus software and firewalls. These are used to protect the whole payment system. However, using them does not guarantee that there are no viruses or intrusions. For example, a virus may disguise itself as or in legitimate software; and the user may choose to trust and install this software when he/she is asked by the system. In addition, since these measures are all pre-emptive, new viruses and attack techniques may cause damage before security engineers have discovered them and developed effective protections against them.

[5] `http://www.kb.cert.org/vuls/html/search`
[6] `http://nvd.nist.gov/`
[7] `http://www.securelist.com/en/descriptions`

2. Authentication of the customer. Passwords, cards or other tokens are used to identify legitimate customers. For example, when making a payment by using an e-wallet account, the customer will be asked to provide his/her account name and password; when making a card payment, the customer will be asked to provide his/her card and the PIN. However, this does not guarantee the safety of payment accounts. For example, an attacker may steal all necessary personal information to access to a payment account.

3. Authentication of the merchant, the e-wallet provider and the bank. This is achieved by using *https* which is based on PKI. The customer's web-browser will display a verified logo (e.g. a lock symbol or a glowing bar on the browser) when browsing an *https* protected web-site. This is only effective if the customer knows the correct name of the web-site and the name of the issuer of the key certificate. If a complacent customer failed to check this information, an attacker may direct the customer to a fake web-site.

Based on the above assumptions of existing control techniques and the capability of threat-sources (attackers), we discover that the amount of potential vulnerabilities is high, to simplify our discussion, we only select and discuss a few distinct vulnerabilities. Since these are all exploitable vulnerabilities, we present them as attacks.

Three attack categories are selected to evaluate payment solutions: credential harvesting, man-in-the-middle and man-in-the-shop. Credential harvesting (CH) represents attacks that are used to steal user credentials which can be reused in attacker-initiated payments. Man-in-the-middle (MITM) represents attacks where the attacker sits in the communication channel between the customer and the merchant and then commits attacks by manipulating messages. Man-in-the-shop (MITS) represents attacks initiated by an insider who has access to the resources in the merchant payment system or the e-wallet payment system. Note that although techniques involved in MITS attacks may sometimes overlap with those of MITM attacks, to simplify our discussion, we list MITS attacks separately from MITM attacks.

Figure 4.3 shows the relationship between attacks and system assets. It also shows the range of our discussion; for example, we do not discuss MITM attacks against connections that are not connected to the customer; we do not discuss attacks against the banking system.

We use acronyms to represent different attack vectors in the attack graph. Note the prefix "A" stands for "Attack". We use rectangles to represent attack categories and circles to represent the detailed attack techniques. Figure 4.4 on Page 53 shows the attack graph.

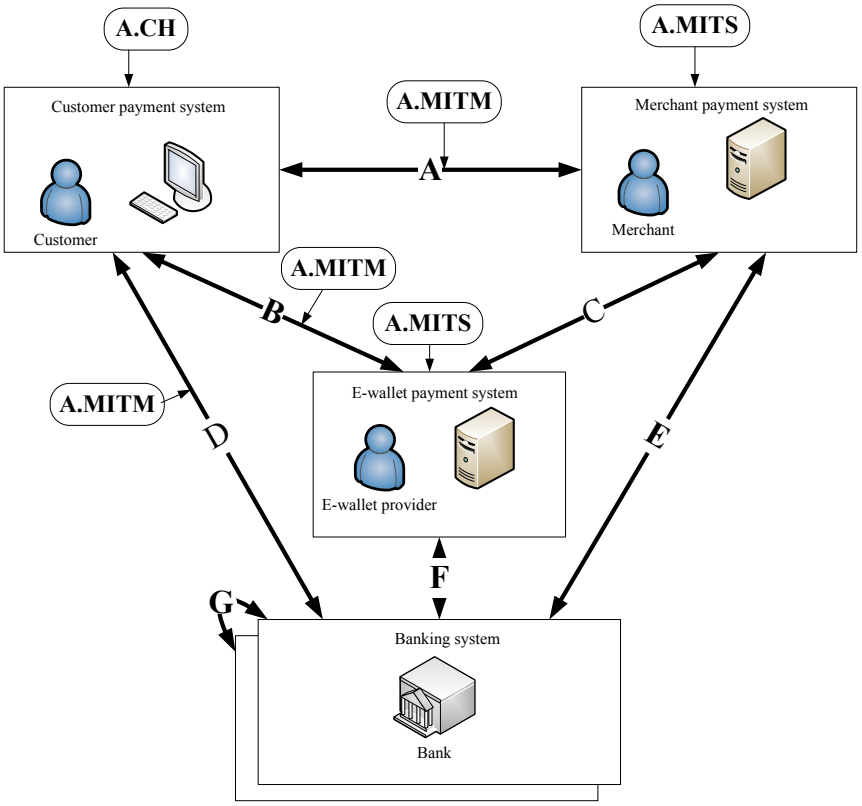


Figure 4.3: The relationships between discussed attacks and system assets.

4.5.1 Credential harvesting (A.CH)

Credential harvesting attacks can be mounted either by social engineering or by installing malware onto the customer's PC. Common techniques are: stealing credentials from PCs; creating fake web-sites and directing traffic to fake web-sites; and social engineering.

• Stealing credentials from PCs (A.SCP)

PC malware threatens all operations made on PCs. If a PC has been compromised by malware, little can be done to secure online payments made on that PC.

A common technique is Key Logging (A.KL). The attacker can install a key-logger onto the user's PC where inputs from the user are recorded and sent to the attacker. In this way, information like the user's account numbers, card numbers, safety questions' answers, PINs and passwords can be harvested and reused. Advanced key-loggers can record PC screens and upload screen shots to attackers. Techniques used to prevent key logging, such as using mouse clicks to input passwords, can be defeated by this attack.

• Creating fake web-sites and directing traffic to fake web-sites (A.CFW)

Creating fake web-sites is usually called phishing: the attackers wait for the customer to visit a fake website and enter their login details. Phishing is one of the most common attacks on the Internet. Various attack techniques have been developed to direct customers to visit these fake websites. We select and discuss a few distinct attack techniques in this section:

1. Using misleading web-site names (A.MIS). Attackers create fake websites, the names of which are visually similar to genuine web-site names. This technique is often called the homograph attack. Examples can be found in [150]. In addition, attackers also exploit customers' typos when entering the web-site names. Examples can be found in [158].

2. Modifying host file (A.MHF). Malware like Qhost [12] can modify the host file on the customer's PC. The customer will be directed to the attacker's fake website, even though he/she has entered the correct name of his/her online banking web-site.

3. DNS poisoning (A.DNS). This is also called DNS cache poisoning. The attacker inserts fake entries into the cache of a recursing DNS server. If this attack is successful, the customer will receive a fake IP address and will be directed to the fake website.

4. Evil Internet access (A.EIA). This consists of three kinds of attacks: evil wireless access point, evil proxies and evil router. For example, the attacker sets up a wireless access point and attracts customers to connect to it. Connections to the evil wireless access point are manipulated by the attacker. The customer using the evil wireless access point can be tricked into visiting a fake web-site which can steal his/her credentials. In public areas, this technique can be effective because customers can be attracted by the "free" Internet services provided by the evil wireless access point. This technique is often used to download malware onto the user's PC. Although it can be defeated by using SSL protection, we have observed that many online banking and online shopping web-sites do not use SSL protection initially; for example, `http://www.barclays.co.uk`, `http://www.hsbc.co.uk` and `http://www.ebay.co.uk`. These provide opportunities for attackers. For example, an attacker can set up a fake banking web-site which distributes malware. The installed malware can then be used to defeat SSL protection. Evil proxies and evil routers work in a similar way.

• Social engineering (A.SE)

Social engineering is a technique often used to convince customers to do things that can expose their credentials. This technique is often used together with other attacks.

In this section, we select and discuss shoulder surfing, email spamming, and phone fraud.

- Shoulder surfing (A.SS). The attacker observes and records the credentials entered by the user. A typical example is when the user is withdrawing money from an ATM and the attacker watches and records the digits of the PIN entered by the user over his/her shoulder. A more advanced attack is to install a secret camera in a position where the user inputs can be recorded. This can be a serious attack which can constantly harvest PINs and passwords. The worst case is when the attacker can gain access to the in-shop surveillance system.

- Email spamming (A.ES). Email spamming is one of the most commonly known techniques of online social engineering. It is often used in combination with credential harvesting techniques. The attacker sends emails to the customer to lure him/her to visit fake web-sites or to download malware. Emails are often disguised as official letters from banks by modifying the email header. Email contents often include alert messages convincing the customer of the urgency of clicking the bogus hyper-link included in the email. Examples can be found in [56].

- Phone fraud (A.PF). Phone calls are often assumed to be secure, and they are frequently used in online payments when the bank calls the customer to verify payments. Attackers often exploit this assumption and use phone calls to commit fraud against customers. For example, the attacker may pose as a member of a bank or a payment company and ask the customer to authenticate himself/herself by providing his/her credentials. Banks compound this problem: they make phone calls to their customers and ask various details for verification without proper notice. The customer often has little information to verify that phone calls are from banks. The voice and the phone number may not be recognised and, in the worst case, the phone number may not be displayed when it is made from a switchboard. More examples can be found in [71, 151].

4.5.2 Man-in-the-middle (A.MITM)

MITM attackers actively manipulate messages exchanged between the user and the merchant. For example, the attacker may modify, block, insert, replace or replay any messages sent and received by the customer. In contrast, attacks discussed in the previous section can be considered as "passive" attacks.

MITM attacks are among the most dangerous attacks. Traditional MITM attacks are often made by session hijacking. The attacker intercepts data packets of an established session; he/she then injects his/her own data packets into the session by pretending to be one of the communicating parties. Common techniques involve IP spoofing and TCP sequence number prediction. More details about this attack can be found in [75, 78]. Session hijacking is often used to guess session IDs or steal session

ID cookies. If successful, the attacker can impersonate a legal party. Unless the entire session is encrypted, session hijacking is a potential threat to electronic payments.

The introduction of SSL has made traditional session hijacking attacks ineffective. However, more sophisticated attacks have been developed to defeat SSL protection. In this section, we select and discuss SSL hacking, man-in-the-browser attack and phone hacking as attacks that are capable of defeating security when even strong encryption is used.

Note that MITM attacks often involve other techniques discussed in the previous section; to simplify our analysis, we do not display these on the attack graph.

• SSL hacking (A.SSL)

SSL was supposed to mitigate MITM attacks of TCP/IP networks. However, attack techniques against SSL (usually humans' use of SSL) have been published [144] and demonstrated [126]. More discussion about attacks against SSL can be found in [45, 32].

Such attacks often exploit the weakness of the customer: the customer may either overlook the warnings on the browser, misconfigure the web-browser, or "forget" to check the issuer of the certificate. Research has indicated that this problem originates from the weak binding of PKI certificates [63]. Indeed, the PKI certificate gives only limited information of a name. And the customer often lacks the proper knowledge to manage and verify certificates.

• Man-in-the-browser (A.MITB)

A Man-in-the-browser (MITB) attack is generally initiated by trojans embedded in the users' browser, for example, Zeus, Adrenaline, Sinowal and Silent Banker [4], which can then manipulate the online payment session in real-time and carry out legitimate online payments. Therefore, solutions that rely on or use the security provided by web browsers to display the order information on PCs are vulnerable to MITB attacks. The MITB technique is an active way of web-page spoofing where the attacker manipulates a payment session in real time.

• Phone hacking (A.PH)

We focus on attacks that can intercept and modify contents received via telephony. These can be achieved by installing malware on mobile phones or by hacking the mobile phone data connections. For example, trojans [39] are found on mobile phones that can hijack SMS communication. Many techniques are reported [114, 110] of hijacking mobile phone data connections.

A MITM attacker can exploit this vulnerability to defeat two-factor authentication deployed by banks. For example, when a bank sends an SMS containing an

authentication code to its customer to verify a payment, the attacker intercepts it and replace it with its own code. This also affects the use of SMS as empirical channels.

4.5.3 Man-in-the-shop (A.MITS)

Man-in-the-shop attacks include what are often termed as insider attacks. According to The CERT Insider Threat Center[8], these attacks are responsible for major financial losses. A report of insider attacks in the banking and finance sector reveals the threshold of committing such attacks is low and financial losses to these attacks are high [111]. In our case studies, a MITS attacker is not limited to existing or former employees, it can be anyone with access to the merchant's or the e-wallet company's resources; for example, an attacker may tamper with a payment terminal or hack into the merchant's or the e-wallet company's database.

We observe that a MITS attacker may take advantage of four factors that the customer usually trusts: people, payment devices, data and environments. We analyse these four factors in the following sections.

• Malicious people (A.MP)

People we usually trust, such as merchants, shop staff and service providers, may not act responsibly, or, in the worst case, may become attackers.

For example, incidents of restaurant workers using skimmers to clone customers' credit cards have been frequently reported [132, 131]. The worker takes away the customer's card and uses a skimmer device to clone the card in secret: the customer believes that the worker is using his/her card to pay for his/her dinner. The input of their PIN as well as the customer's signature can be recorded by the worker as well.

It is important for the customer to manage access control of his/her bank cards. The best practice may be that the customer never hands out his/her cards to anyone.

• Malicious payment devices (A.MPD)

For example, when making a card payment in a shop, the challenge is that the customer must use his/her card on a payment device that belongs to others. This can become a weakness in the payment process when the attacker can get access to the machine.

The attacker exploits the vulnerabilities of payment devices; for example, point-of-sale (POS) machine, to clone customers' cards or steal card data. A POS machine can either be tampered with or replaced by a fake one. Researchers at Cambridge University have developed effective tools to tamper POS card readers [116]. Incidents of these have been reported [92].

[8] `http://www.cert.org/insider_threat/`

- **Insecure data management (A.IDM)**

The PCI DSS standard allows the merchant to keep records of customers' card data [10]; for example, records of these can be used in charge back services. Online merchants also keep records of card data which can be reused when the customer logs onto their web-site. This – the aggregation of customers' sensitive data on the merchant's side – attracts attacks worldwide. In 2011, server breaches at well-known companies, like Sony, RSA Security, Epsilon and Valve, have been reported [18]. Millions of copies of customers' bank cards may have been stolen. Such incidents lead to concerns regarding organisational security: companies, especially when their size is small, may not be capable of maintaining high data security on their servers [105].

- **Malicious Environments (A.ME)**

Environments are where payments take place. These may include the physical environment in a shop, or the virtual environment of an application which receives payments. For example, we have discovered that a large shop in Oxford had a camera recording PIN inputs by their customers. This is an obvious failure of security design which can be exploited by attackers. Customers in this case would never check that the shop is spying on them.

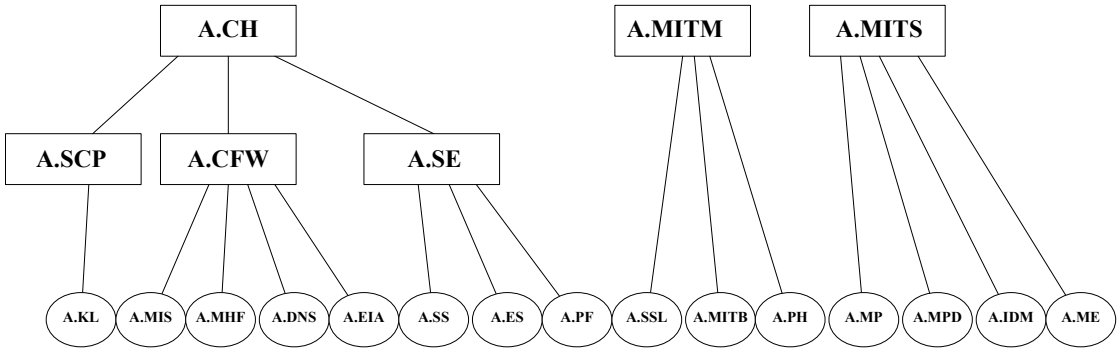


Figure 4.4: Attack graph.

4.6 Quantifying risks

4.6.1 Likelihood Determination

According to SP 800-30, the likelihood is determined by: (A) the attacker's motivation and capability; (B) the effectiveness of controls. The guidance of determining the likelihood is as follows:

IF Motivation is high AND Control Effectiveness is low, THEN Likelihood is High.

IF Motivation is high/medium AND Control Effectiveness is medium, THEN Likelihood is Medium.

IF Motivation is low OR Control Effectiveness is high, THEN Likelihood is Low.

Values assigned to Low, Medium and High may vary in different analysis. To simplify our analysis, we use only three values according to the example used in SP 800-30: High = 1.0; Medium = 0.5; Low = 0.1. Note these values are somewhat subjective.

In general, the motivation behind any attack against payment is high. Therefore we only focus on Control Effectiveness. Control Effectiveness in payments may consist of two factors:

1. The effectiveness of existing controls. According to our assumptions on existing control techniques, using these techniques cannot eliminate attacks. And we have indicated that protections are not always pre-emptive. For example, anti-virus software is only effective after the new virus is detected and studied. Therefore there is a delay between the occurrence of the attack and the protection becoming effective against the attack. The risk is certainly higher if the technique and the form of an attack keep changing.

2. The popularity of the attack. Since protections are not always pre-emptive, the more popular an attack is, the more damage it is likely to cause. For example, the more popular an attack is, the more pervasive it is likely to be; in addition, the development speed of the variants of the attack is likely to be higher. The popularity is measured by referring to the malware statistics and our observations.

Table 4.5 gives the guideline likelihood values of different attacks.

Table 4.5: The guideline of attack likelihoods.

Attack	**Likelihood**	**Notes**
A.KL	0.5	Keylogging is a common attack [157]. Protection: High. Popularity: High. Overall: Medium.
A.MIS	0.1	Protection: High. Popularity: Low. Overall: Low.
Continued on next page		

Table 4.5 – continued from previous page

Attack	Likelihood	Notes
A.MHF	0.1	Trojans that modify host files are widely reported. For example, Qhost is a well-known type of trojan that modifies the host file on windows. This attack is normally effective on Windows only. Protection: High. Popularity: Medium. Overall: Low.
A.DNS	0.1	DNS poisoning requires sophisticated techniques. Most current DNS servers often have been designed to be secure against poisoning attacks. Protection: High. Popularity: Low. Overall: Low.
A.EIA	0.1	The cost of this attack is high and often requires physical access to the access point, for example, the evil wireless access point has to be set up by the attacker manually. Using SSL can detect this attack. Protection: High. Popularity: Low. Overall: Low.
A.SS	0.1	Shoulder surfing requires the physical on-site presence of the attacker or it may require the installation of image recording devices. The cost of this attack is high. Protection is only effective when the customer is cautious. Protection: Medium. Popularity: Low. Overall: Low.
A.ES	0.1	Email spamming is a common social engineering technique. Google, Yahoo and other well-known email service often provide effective spam filtering services and they display warnings if fake email headings or bogus hyper-links are detected. Protection: High. Popularity: Medium. Overall: Low.
A.PF	0.1	The cost of phone fraud is high. The protection is only effective when the customer is cautious. Protection: Medium. Popularity: Low. Overall: Low.
A.SSL	0.1	Brute-force attack against SSL is difficult. Successful attacks often exploit human complacency. Legitimate systems often employ sufficient intrusion detection mechanisms to warn the customer if an untrustworthy certificate is to be installed. Protection: High. Popularity: Low. Overall: Low.
A.MITB	0.5	It is achieved by installing trojans on the PC. Statistics from a recent report show Trojans against browsers are common among malware attacks [118]. Protection: High. Popularity: Medium. Overall: Medium.

Continued on next page

Table 4.5 – continued from previous page

Attack	Likelihood	Notes
A.PH	0.1	The amount of mobile malware is growing fast. A majority of these malware aim at spying data communication. But the popularity of mobile malware is currently limited to two mobile platforms according to [118]. Protection: High. Popularity: Medium. Overall: Low.
A.MP	0.1	The cost of carrying out this attack is high because of the requirement of physical on-site presence. Detecting it can be difficult because the surveillance of behaviours of merchants, shop staff and service providers may not be sufficient. Protection: Medium. Popularity: Low. Overall: Low.
A.MPD	0.1	It often requires physical access to hack payment devices and the cost of this attack is high. Because payment devices such as till machines are not frequently checked and many are not tamper-proof, detecting this attack can be difficult especially in small shops. Protection: Medium. Popularity: Low. Overall: Low.
A.IDM	0.1	Databases of large companies or organisations often use strong security. Attacks to steal data require (A) sophisticated hacking techniques and (B) knowledge of vulnerabilities exposed in the payment system's data flow. Protection: High. Popularity: Low. Overall: Low.
A.ME	0.1	Controlling the payment environment is difficult unless the attacker is an insider; for example, an attacker has to get access to the shop and modify or install equipments without being detected. Protection: High. Popularity: Low. Overall: Low.

4.6.2 Impact Analysis

SP 800-30 recommends three metrics to analyse attack impacts: integrity, availability and confidentiality. There are three impact magnitudes: High, Medium and Low. To measure impacts, they recommend using the following three factors: (1) loss of assets or resources; (2) loss of organisation's mission, reputation or interests; (3) human injuries or loss of human lives. However, in payments, we are more concerned about the financial loss and there is unlikely any human injury or loss of human lives. In addition, in payments, the number of customers that may be affected by an attack is useful in measuring the impact. We therefore modify the three factors of measuring impact magnitudes in order to use them in the evaluation of payment solutions:

1. High. The loss of integrity, availability or confidentiality (1) may result in highly costly financial loss of the merchant or the customer; (2) may seriously undermine the merchant's services and reputation; (3) may affect some or all customers.

2. Medium. The loss of integrity, availability or confidentiality (1) may result in costly financial loss of the merchant or the customer; (2) may undermine the merchant's services and reputation; (3) may affect a single or some customers.

3. Low. The loss of integrity, availability or confidentiality (1) may result in the financial loss of assets of the merchant or the customer; (2) may noticeably undermine the merchant's services and reputation; (3) may affect only a single customer at a time.

To simplify the analysis, we use only three values according to the example used in SP 800-30: High = 100; Medium = 50; Low = 10. These values are also subjective. Before our discussion, we specify a few general guidelines as follows:

1. The loss of the credentials may result in a maximum loss of all the money in the payment account. The impact to an individual is therefore maximum. The overall impact is then measured by the number of possible victims.

2. The MITM attacks may result in a maximum loss allowed in a single payment. These attacks can also be used to steal credentials. Therefore the overall impact is measured by the number of potential victims and the number of possible payments that may be affected.

3. The MITS attacks may result in serious impacts to the customer, the merchant and the e-wallet provider. The number of victims can be high and the potential loss can reach the maximum on both sides. Therefore, unless otherwise stated we put the overall impact as high.

Table 4.6 gives the guideline impact values of different attacks. Note it may be necessary to modify these values depending on the details of the examples being studied.

Table 4.6: The guideline of attack impacts.

Attack	**Impact**	**Notes**
A.KL	50	The impact is limited to the user of the infested PC. Overall impact: Medium.
A.MIS	10	The impact is limited to complacent customers only if they made mistakes. Overall impact: Low.
Continued on next page		

Table 4.6 – continued from previous page

Attack	Impact	Notes
A.MHF	50	Similar impact to A.KL's.
A.DNS	50	All customers using the poisoned DNS server will be affected. But only complacent ones who ignore the warnings will use the fake websites. Overal impact: Medium.
A.EIA	50	The impact is limited to customers who choose to use the bogus wireless access point. Overall impact: Medium.
A.SS	10	The impact is limited to customers at a specific location, and only those complacent ones will be affected. Overall impact: Low.
A.ES	10	The number of people who receive email spams is large, but the impact is limited to those who are not only complacent but also choose to follow the instructions in the email spam. Overall impact: Low.
A.PF	10	The impact is simiar to A.ES's except that the number of receivers is smaller. Overall impact: Low.
A.SSL	50	A successful SSL attack may result in the loss of credentials and transaction data on a specific website. The impact is limited to the user (who has installed the fake certificate) of a specific web-site. Overall impact: Medium.
A.MITB	50	An MITB attacker cannot only steal credentials, they also control the payment session in real-time. But the impact is limited to the user of the infested web-browser. Overall impact: Medium.
A.PH	50	A hacked mobile phone affects payments using phone factors. The impact is limited to the user of the hacked mobile phone. Overall impact: Medium.
A.MP	100	The impact is made to all customers served by the malicious people. It may result in the loss of credentials and the immediate loss of money. Overall impact: High.
A.MPD	100	The impact is made to all customers who use the malicious payment device. Overall impact: High.
A.IDM	100	It may result in the loss of credentials of all customers of the payment service. The impact of this attack is the highest among all impacts of attacks listed in this table. Overall impact: High.

Continued on next page

Table 4.6 – continued from previous page

Attack	Impact	Notes
A.ME	100	The impact is made to all customers who are in the malicious environment; for example, in a shop or in front of a till machine. Overall impact: High.

4.6.3 Risk Determination

According to NIST SP 800-30, the algorithm to determine the level of risk is:

$$Likelihood \times Impact = Risk$$

And the risk scale is: High (>50 to 100); Medium (>10 to 50); Low (0 to 10). The followings are the explanations of each risk level and the necessary actions:

1. High. It is dangerous to continue using the existing payment solution. Immediate actions must be taken to improve the security of the existing payment solution to defend against the attack.

2. Medium. The existing payment solution can continue to be used. The security of the existing payment solution should be improved in the near future to defend against the attack.

3. Low. The existing payment solution can continue to be used. Actions to improve the security of the existing payment solution to defend against the attack is optional.

4.7 Case study

We discuss and evaluate a few selected online payment solutions. Materials of our case study are obtained from our investigations. We focus on evaluating their security attributes as well as the complexity of using them. We use D to represent the number of digits/characters of input, M to represent the number of digits/characters a customer has to remember between transactions, and P to represent the number of phases[9] involved. Identified attack vectors are presented by using the attack graph, and results of risk and complexity evaluation are presented by using tables.

[9]The word phase in this section represents the stage the customer needs to go through; for example, when making a payment on an online shopping web-site, the customer may need to be directed to a different web-site to complete the payment. In this case, the customer needs to go through at least two phases.

4.7.1 Web-based card payments

A report from UK Payments Administration (previously known as Association for Payment Clearing Services (APACS)) indicates a decrease in card-not-present fraud in 2009 [1], the first decrease since 1999. They attribute this achievement to the use of MasterCard SecureCode and Verified by Visa, the 3-D Secure (3DS) protocol[10] for authenticating online card transactions (customer to merchant), which is entering the card number and password into an inline-frame form or a pop-up form on the merchant's website to authorise the online card payment. However, this protocol has serious security flaws vulnerable to phishing, MITM attacks and malware [115]. And it is suggested in [115] that by using a trusted device with USB connection and display we can address those problems in the long term. A different version of 3DS protocol is implemented by French bank Société Générale.

A typical 3DS payment involves the input of a 16-digit card number, a 4-digit expiry date and a 3-digit security code onto the merchant's website, and 3 or more characters selected from the password (its minimum length is 6 characters) onto a 3DS protocol web form. Figure 4.5 shows the attack graph of 3DS protocol. Table 4.7 shows the evaluation results of 3DS protocol under the measures set out above.

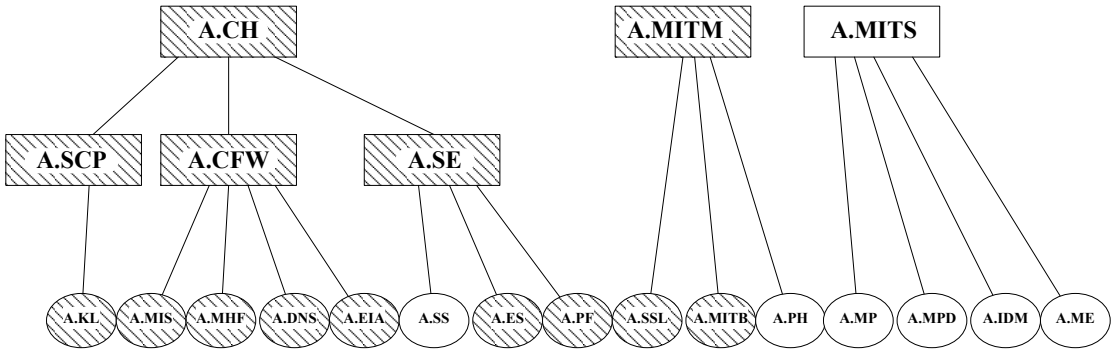


Figure 4.5: The attack graph of 3DS protocol.

4.7.2 E-wallet payment services

E-wallet payment services are a popular form of online payment; well-known examples are Paypal[11] and Google check-out[12]. We choose to evaluate Paypal in this section.

[10] One required to accept this protocol in order to pay online by using their debit cards. We note that this can be time-consuming and inconvenient.

[11] `https://www.paypal.com`

[12] `https://checkout.google.com`

	Attack	Likelihood	Impact	Risk
Security	A.KL	0.5	50	25
	A.MIS	0.1	10	1
	A.MHF	0.1	50	5
	A.DNS	0.1	50	5
	A.EIA	0.1	50	5
	A.ES	0.1	10	1
	A.PF	0.1	10	1
	A.SSL	0.1	50	5
	A.MITB	0.5	50	25
			Overall risk:	73

Complexity	D	M	P
	26	≥ 6	1

Table 4.7: The evaluation results of 3DS protocol.

The customer first registers his/her bank card and bank account details on the e-wallet payment website, and then obtains an e-wallet account.

Online merchants will receive payments from e-wallet companies, therefore only the account from the e-wallet company is exposed. In this way, the e-wallet company provides an effective shield of the customer's bank cards and accounts. It is an effective mechanism of dealing with the high volatility of online financial transactions where disputes may happen frequently.

A general process of using Paypal is as follows:

1. The customer presses the payment button on a web-page of an online shop.
2. The online shop redirects the customer to the Paypal web-page.
3. The customer logs in and verifies the order information. If the customer has already logged in, then he/she only needs to verify the order information.
4. The customer presses the confirm button and Paypal processes this payment.
5. The customer is redirected back to the online shop web-page which displays the payment results.

We assume the customer has already logged into Paypal, therefore the value of D is 0 when making a payment to an online merchant.

When making a peer-to-peer payment through Paypal, the customer needs to enter the payee's email address or mobile phone number. We assume the customer uses the mobile phone number of which the length[13] is 11 digits, therefore the value of D is 11 in this case.

[13]We assume the mobile phone number is a UK phone number.

Although e-wallet companies like Paypal provide protection for customers' bank cards and accounts against online merchants, payment operations are made on PCs. So credential harvesting and MITM attacks are still possible. In addition, according to the analysis in Section 4.5, there are risks to store data on servers of e-wallet companies, therefore it cannot eliminate MITS attacks. Figure 4.6 shows the attack graph of Paypal. Table 4.8 shows the evaluation results of Paypal.

Note that, theoretically, the impact of losing credentials of e-wallet accounts should be smaller than the impact of losing credentials of bank accounts. This is, however, difficult to measure in practice because it depends on the actual amount of money available in the payment account. Therefore we make no change to the impact values provided in the impact evaluation guidelines in Table 4.6.

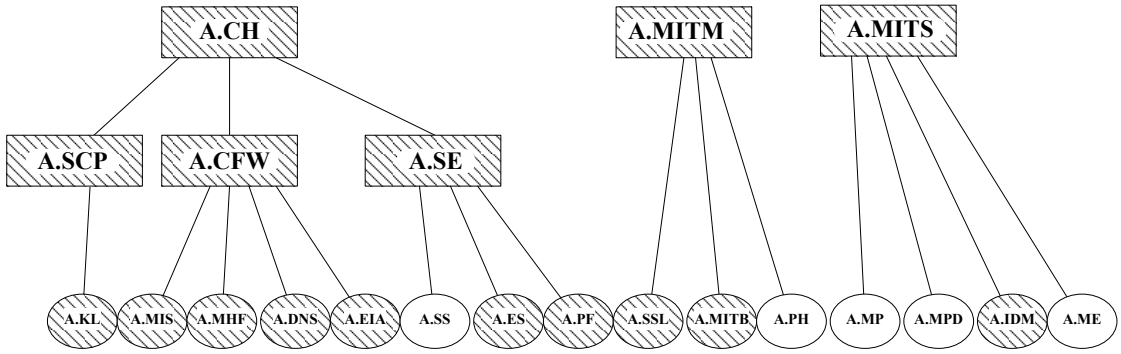


Figure 4.6: The attack graph of Paypal online payment services.

4.7.3 Online banking

• Card Authentication Program

In 2007, banks in the UK started to release large numbers of Card Authentication Program (CAP) readers. Their main purpose is to use the existing EMV[14] protocol to provide two-factor-authentication (PIN and bank card) for online payments. However, the multiple inputs required for a payment are laborious. For example, when you use the NatWest CAP card reader to make a single payment to a person, you need to:

P1. Enter the payee's name, 6-digit sort code and 8-digit account number, together with the payment reference and the amount of money on the online banking website on the PC;

[14]EMV stands for Europay, MasterCard and VISA.

	Attack	Likelihood	Impact	Risk
Security	A.KL	0.5	50	25
	A.MIS	0.1	10	1
	A.MHF	0.1	50	5
	A.DNS	0.1	50	5
	A.EIA	0.1	50	5
	A.ES	0.1	10	1
	A.PF	0.1	10	1
	A.SSL	0.1	50	5
	A.MITB	0.5	50	25
	A.IDM	0.1	100	10
			Overall risk:	83

	D	M	P
Complexity	11 or 0	≥ 8	3

Table 4.8: The evaluation results of Paypal.

P2. Bring out the card reader and insert the card:

1. Press the "Response" button, enter the 4-digit PIN and then press the "Enter" button;
2. Enter the 8-digit authorisation number (the last four digits are the last four digits of the payee's account number) copied from the web-page and press the "Enter" button;

P3. Type the 8-digit code displayed on the card reader onto the online banking website on the PC.

For such a transaction, a customer needs to enter a minimum of 34 digits, including 4 digits of PIN. In this case, $D = 34$, $M = 4$ and $P = 3$. In another example, using the Barclays' CAP card reader requires slightly more input by replacing the data in Step (2) in Phase 2 with the account number of the payee and the amount of money.

Since the input of a PIN and the reading of card details are carried on the CAP card reader, the malware attack on the PC can be isolated from the device. And by including the authorisation number in the computation of generating the response, the passive phishing attack cannot effectively reconstruct another transaction by reusing the code; but since there is no freshness, and transaction information such as the amount of money included in the input, it is vulnerable to the Man-in-the-Middle (MITM) attack [61]. The Barclays' CAP card reader is better than NatWest's in this case, but the bank still can not check the freshness of the received response. An example in [61] shows a MITM attacker is able to exploit this vulnerability by using a corrupted Chip&PIN terminal. However, if the web-browser is compromised, the

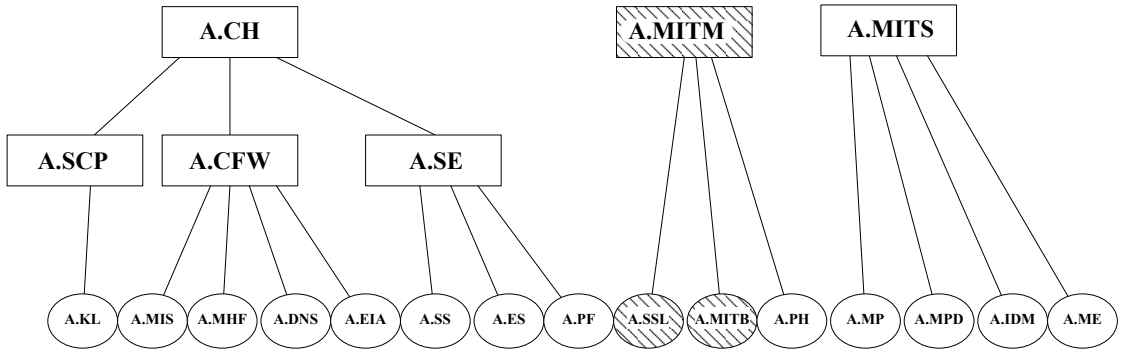


Figure 4.7: The attack graph of CAP.

attacker can carry out a MITB attack without using a tampered terminal. Figure 4.7 shows the attack graph of CAP. Table 4.9 shows the evaluation results.

Security	Attack	Likelihood	Impact	Risk
	A.SSL	0.1	50	5
	A.MITB	0.5	50	25
			Overall risk:	30
Complexity	D	M	P	
	34	4	3	

Table 4.9: The evaluation results of CAP.

Figure 4.7 shows that the security of using dedicated payment devices is significantly better than that of using PCs: we can isolate our credentials (stored on cards) from PCs and therefore we can reduce threats from PC malware.

However, in practice the data of a complete set of order information may be significantly larger than 34 digits. The human effort of inputting all the information required is too large for complete security to be obtained. This human effort can be replaced if we have an authentic electronic channel between merchant and the card reader: all the required information can be transmitted electronically on this channel. All the customer needs to do is to check necessary parts and authenticate his/her own identity from existing solutions. In this way we can reduce the complexity of using card readers.

HSBC views this two-factor authentication method as too inconvenient for their customers; it is, therefore, one of the very few banks in the UK that has not adopted CAP. Instead, it is reported that it intends to use an OOB technology provided by Authentify to authenticate the customer to the bank [22]. Barclays attempted to

move customers to a similar technology for logging on to the banking web-site but was found by its users unpopular, and then to allow previous password-based mechanism.

It is recognised that a secure electronic connection and a secure display are necessary to solve the flaws of CAP and 3DS protocol. The lack of either one of the two can result in possible attacks.

It is worth mentioning that similar devices were developed a few years ago. One example is provided by those devices described by the EU Electronic Signature Directive, which are used to provide digital signatures in online payments. A short introduction can be found in [149]. The problem that prohibits these devices from being available to ordinary customers is their expense: a typical device may cost over 100 US dollars [61]. Therefore reducing the cost is critical.

• Online banking by mobile phone numbers

The Bank of Communications is one of the largest banks in China. It has implemented a new banking system which allows the customer to make payments using telephone numbers.

The customer first registers his/her mobile phone number with the bank which will bind this number with the customer's bank account. When making a payment, the customer only needs to input the payee's phone number and name, and the payment amount. When the bank receives the payment command from the customer, it will send a query to the database to find out the payee's complete bank account details. The bank will then create a payment command including the payee's bank account details and send it to the underlying banking system.

The binding of the mobile phone number and the bank account virtually converts the phone number and the name into a simpler bank card, which has the following benefits:

1. The telephone number can replace the bank card number as the identification information. It is easier to remember, transfer and input a phone number than a bank card number.

2. The payee's bank card number and bank account number are not disclosed, which protects the payee's privacy.

The telephone number is usually 11 digits, and the customer completes the payment on the same web-page, therefore we put $D = 11$ and $P = 1$. Figure 4.8 shows its attack graph. Table 4.10 shows its evaluation results.

Figure 4.8 shows there are no significant improvements in security when using phone numbers in online payments. Table 4.10 shows that the main threats come from the use of PCs where various malware can steal credentials input by the user. We also notice that binding telephone numbers with payment accounts can significantly reduce

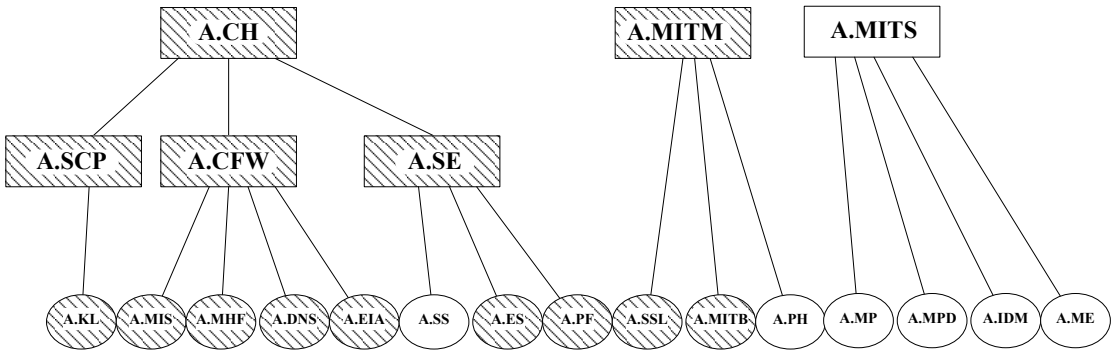


Figure 4.8: The attack graph of the online banking solution of Bank of Communications.

	Attack	Likelihood	Impact	Risk
Security	A.KL	0.5	50	25
	A.MIS	0.1	10	1
	A.MHF	0.1	50	5
	A.DNS	0.1	50	5
	A.EIA	0.1	50	5
	A.ES	0.1	10	1
	A.PF	0.1	10	1
	A.SSL	0.1	50	5
	A.MITB	0.5	50	25
			Overall risk:	73
Complexity	D	M	P	
	11	0	1	

Table 4.10: The evaluation results of the online banking solution of Bank of Communications.

the complexity of making online payments. This gives a possible way of improving usability that could can be used in other payment solutions.

• Out-of-band technology

OOB technology is pervasive in the financial world. For example, CAP requires the customer to enter the challenge value displayed on the bank's *https* web-page on the CAP card reader, in which the manual reading and entering of the challenge value by the customer constitute an OOB channel between the bank and the CAP card reader. Customers entering their card details when completing online shopping is a typical OOB way of establishing a connection between the web-site and the card (although,

only one way).

The following two cases are examples of the current in-use OOB technologies.

Authentify[15]. When the customer starts to pay online, the Bank's Server (BS) sends its telephone number to Authentify's Server (AS). Then AS makes a phone call to the customer and meanwhile AS sends a Random Number (RN) to BS. After the RN is received BS forwards the RN to the customer's PC screen. The customer then enters the RN on his phone; AS checks whether the received RN matches with the original one and, if successful, AS informs BS that the customer is authenticated by AS. The OOB channel here is telephony.

The phishing attack against Authentify fails because it cannot obtain the RN from the PC. The MITM attack can be successful by relaying the transaction data. Because the RN gives no information of the actual transaction the customer is authorising, the customer would falsely send back the RN to AS.

A further vulnerability of this approach is that it depends crucially on the security of the telephone network. Consider, for example, the attack on mobile data connections [114]. Figure 4.9 shows the attack graph of Authentify. Table 4.11 shows the evaluation results of Authentify.

According to our test of Authentify's online transaction demo[16], it only requires an input of the card number (and the mailing address). However, the major difference to a 3DS protocol is that you need to input an extra 6-digit code on your mobile phone, together with a voice recording of your name and the words "I approve this transaction". T represents the effort of voice recording in Table 4.11.

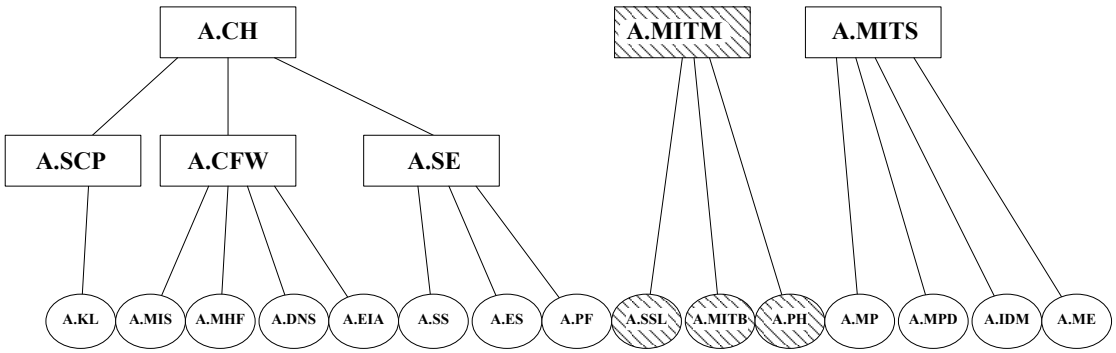


Figure 4.9: The attack graph of Authentify.

Société Générale. This bank implemented the 3DS protocol, but differing from UK banks, it requires the customer to enter an additional 6-character authentication

[15] `http://www.authentify.com/`

[16] The test we did is on 11 Feb. 2011. Demos can be found on Authentify's website.

	Attack	Likelihood	Impact	Risk
Security	A.SSL	0.1	50	5
	A.MITB	0.5	50	25
	A.PH	0.1	50	5
			Overall risk:	35

	D	M	P
Complexity	22+T	0	2+T

Table 4.11: The evaluation results of Authentify.

code received from the bank via SMS or phone call [13]. Although it looks similar to Authentify's solution, their security attributes are different. The code is entered on the web-page and a MITM attacker can reuse the code in another transaction before it expires.

However, the additional phone factor does reduce the impact of "passive" attacks: the attacker has to carry out the attack in real time. Therefore we adjust the impact of credential harvesting attacks to low.

Figure 4.10 shows the attack graph of Société Générale. Table 4.12 shows its evaluation results.

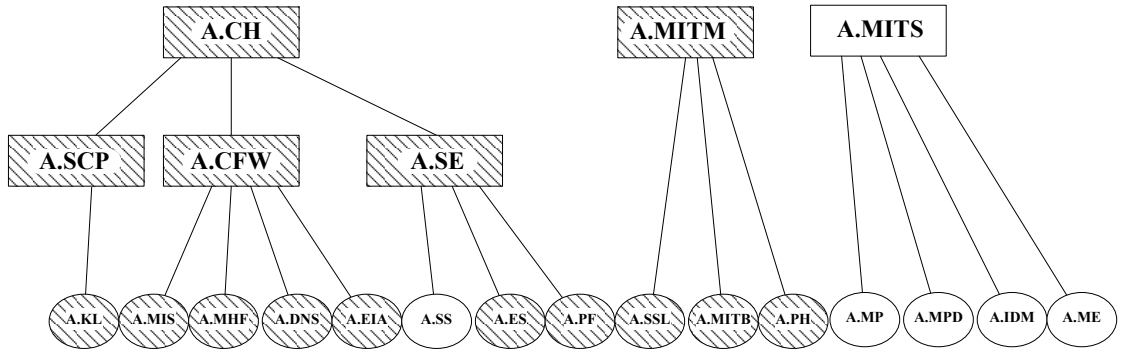


Figure 4.10: Attack graph of Société Générale.

Table 4.12 shows that the improvements in security are smaller than the improvements achieved by Authentify. The example of Authentify shows that if we can reverse this process by entering the authentication code received via PC on the mobile phone, we can eliminate passive attacks: only active MITM attackers can succeed by displaying the relayed authentication code on the PC. The human effort required in this case is 6 more characters than 3DS protocol.

	Attack	Likelihood	Impact	Risk
Security	A.KL	0.5	10	5
	A.MIS	0.1	10	1
	A.MHF	0.1	10	1
	A.DNS	0.1	10	1
	A.EIA	0.1	10	1
	A.ES	0.1	10	1
	A.PF	0.1	10	1
	A.SSL	0.1	50	5
	A.MITB	0.5	50	25
	A.PH	0.1	50	5
			Overall risk:	46

	D	M	P
Complexity	32	≥ 6	1

Table 4.12: The evaluation results of Société Générale.

4.7.4 Other card payments

Lakala[17] is a payment company which distributes a large number of unmanned machines in different cities. Such a machine provides many online payment services by using bank cards or credit cards, for example, to pay utility fees, pay credit card debt, transfer money, or check bank accounts. In this way, they claim that the customer can enjoy improved convenience to use their cards to pay others by walking to the nearest Lakala machine. Note that this case falls out of our definition of online payments. However, we can treat this as a card reader[18] connected to a PC located in a public place. This interpretation allows us to use it as an example of showing how new designs may expose new vulnerabilities.

According to the analysis in Section 4.5, we discover high risks of MITS attacks in this service. These machines are often unattended and positioned in public areas. Attackers with the necessary skills can get access to these machines and tamper with them. Clearly the threshold of committing such attacks is lowered in comparison to committing attacks against machines in shops.

Because we have not tested this method in practice, there is no analysis of their complexity. It raises our concerns because new services like this were once praised as an effective design in improving the usability of card payments. Potential risks are ignored. Figure 4.11 shows the attack graph of Lakala. Table 4.13 shows its evaluation results.

[17]The analysis is based on their presentation which is obtained during our industry investigation in China.

[18]We assume that the customer has the same level of trust when using the card reader on the Lakala machine as that of using his/her own card reader at home.

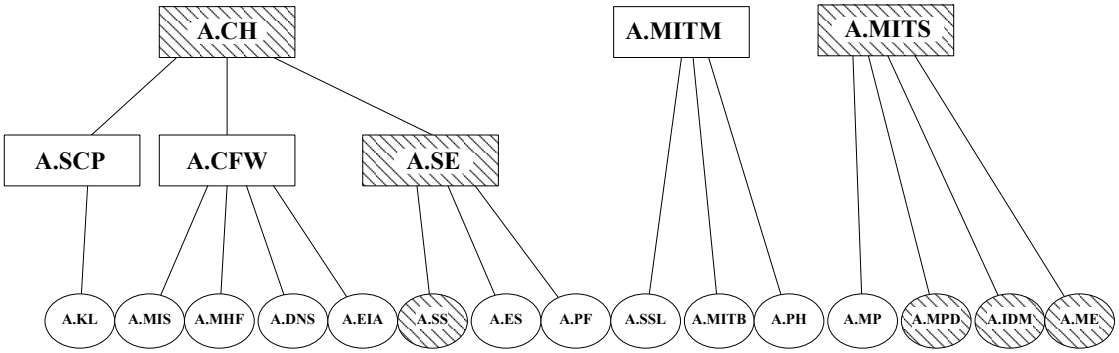


Figure 4.11: The attack graph of Lakala.

	Attack	Likelihood	Impact	Risk
Security	A.SS	0.1	10	1
	A.MPD	0.1	100	10
	A.IDM	0.1	100	10
	A.ME	0.1	100	10
			Overall risk:	31
Complexity	*D*	*M*	*P*	
	N/A	N/A	N/A	

Table 4.13: The evaluation results of Lakala.

4.8 Requirements for our new payment solution

We have analysed seven different online payment solutions. To better understand the requirements for a new solution, we compare the results of previous analysis.

Figure 4.12 on Page 72 gives a comparative view of the total risks of online payment solutions discussed. We observe the following facts:

1. Using e-wallet payment services does not technologically improve security. The introduction of e-wallet payment companies may increase overall risks because of the potential insecure server-side data management.

2. Using phone factor can improve security[19], especially when the authentication code is entered on the mobile phone rather than on the PC because the authentication code is designed to be secret.

3. Using a dedicated card reader provides the strongest security since sensitive data are entered and processed on the card reader, rather than on the PC.

[19]A discussion on comparing PC security and mobile phone security is presented in Section 5.5.

4. Using card readers of others can compromise well-designed security because of the potential MITS attacks.

We therefore conclude the following requirements from the above facts: (i) to create a card reader which is carried and used by the customer himself/herself; (ii) the phone factor can be used when strong security is required; (iii) money is transferred directly from the customer's bank account to the merchant.

Figure 4.13 on Page 73 displays their complexities[20] (we display the value of D). We observe the following fact: using telephone numbers to replace account details can significantly improve usability. This advantage is more obvious when making peer-to-peer payments because (A) the privacy issue is more important in peer-to-peer payments and (B) the payer may not necessarily know the complete account details of the payee. In practice, telephone numbers may not be the only option to replace account details, for example, we can use email addresses, social network accounts, nick names, addresses, domain names or even photos to identify the payee as long as this information is authentic.

We also observe that when making payments to online merchants by using Paypal, no inputs of digits/characters is required if the customer has already logged into his/her Paypal account. No input of digits/characters is a strong advantage; to achieve it we need to solve two challenges:

1. To electronically transfer the order information from the merchant to the customer.

2. To electronically transfer the money from the customer to the merchant.

Both challenges can be resolved if we can establish an electronic connection between the customer and the merchant. Paypal achieves this by integrating its service into online shopping web-sites. For example, when the user clicks the payment button on a web-site, it will send a payment command to Paypal; the customer will be directed to Paypal's web-page where the merchant's details and the payment amount are displayed; the customer then clicks the confirm button which will trigger Paypal to transfer money from the customer's account to the merchant's account. The electronic connection in this case is between the customer's Paypal account and the merchant's website.

We therefore add the following requirement to the existing three requirements: (iv) to create a secure electronic connection between the customer's card reader and the merchant's server. Requirement (iv) is clearly the most important of them all. Using a secure electronic connection also maximises security because a larger set of information can be received, displayed and processed securely on the card reader. And it also provides the opportunity of using an e-cheque to complete the payment, which can eliminate MITS attacks.

[20]Lakala is excluded because we have not tested it.

However, Requirement (iv) is difficult to satisfy. When using Paypal, the user interface is the web-browser on the PC which is already connected to the Internet, and establishing a connection between two web-sites on the web is easy. When using a card reader, the card reader cannot easily be connected to the Internet, and establishing a secure connection between the card reader and the merchant's server is complicated and difficult.

In the following sections, we will discuss details of how to design and implement a new payment solution that achieves Requirement (i) to (iv).

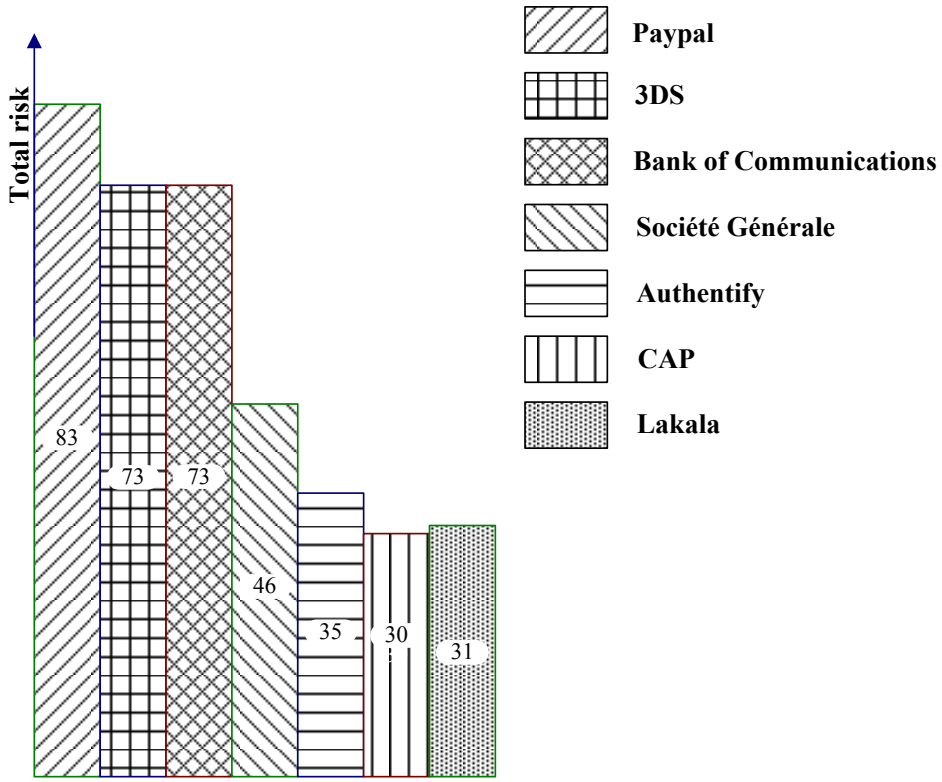


Figure 4.12: A comparison of overall risks of seven online payment solutions.

4.9 Using a HISP

The purpose of using a HISP in online payments is to bootstrap a secure connection between the customer and the merchant (or the bank) before any private information

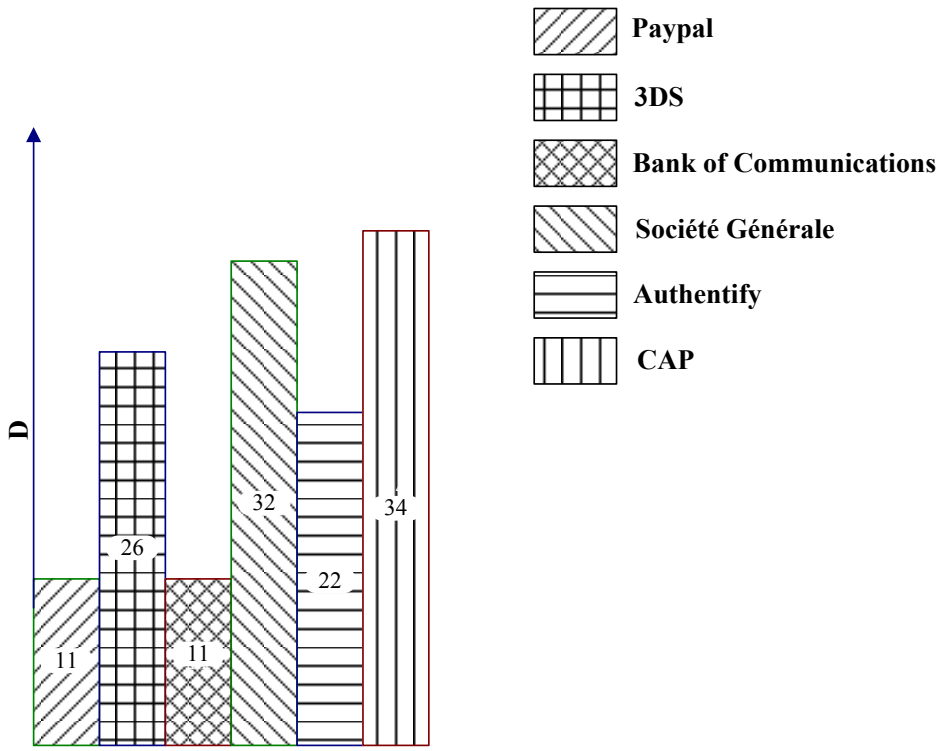


Figure 4.13: A comparison of complexities of six online payment solutions.

is exchanged, such as the customer's payment account details. If we can successfully establish such a secure connection, it will allow the customer to automatically download the order information from the merchant. This can considerably reduce the amount of human effort. The customer will only be required to verify the received details on his/her trusted device, and then authorise the payment by entering his/her PIN or password. We recommend that an e-cheque is generated and sent to the merchant, which aims to reduce the risks of dealing with corrupt or insecure merchants. This technique can be integrated into online banking as well as online shopping, therefore we give two roles in our example: C represents the customer; M represents the bank or the merchant (or its server).

Our solution gives the customer the assurance that he/she is paying the precise instance of the merchant even without knowing the name or ID of the merchant. It operates in the opposite direction and as a forerunner to the usual authentication of customer to merchant. This is achieved by using the reverse authentication method introduced in Chapter 3. This gives both extra security and enables us to make the traditional security goal of authenticating the customer to the merchant/bank easier and more thorough.

The pair-wise protocol introduced in Chapter 2 has been adapted by adding a few fields required in payment. In the protocol, TD represents the trusted device, M represents a merchant, and C represents a customer. Note that this is not a complete payment protocol. It simply sets up security needed to protect the payment.

1. $TD \longrightarrow_N M : \ ID_{TD}, INFO_{TD}, hash(hk_{TD}, ID_{TD}), hash(k)$
2. $M \longrightarrow_N TD : \ ID_M, INFO_M, pk_M, hash(hk_M, ID_M)$
3. $TD \longrightarrow_N M : \ Encrypt_{pk_M}(k), hk_{TD}$
4. $M \longrightarrow_N TD : \ hk_M$

5a $M \longrightarrow_E C \longrightarrow_E TD : \ digest(hk_{TD} \oplus hk_M,$
$(ID_{TD}, ID_M, pk_M, k, hash(k), INFO_{TD}, INFO_M))$

5b TD compares the *digest value* with its own version.

In Messages 1 and 2, k is a session key (a random number) generated by D, it is exchanged by using the uncertified public key pk_M provided by M. $INFO_M, INFO_{TD}$ represent other information that the actual system would require; for example, date and time, part of the order information, etc.

Naturally, if the protocol has proceeded uninterfered with, TD's and M's values will be equal. If, however, an intruder has imposed his own values on the receivers of Messages 1–4, TD and M will not agree on all four parameters. For security, what is important is that they agree on pk_M 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. Note we call the SAS the digest value here.

To maintain an acceptable security and usability, implementers need to examine carefully the use case and the perceived risks between the user and the merchant. A standard [20] given by 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[21].

Figure 4.14 describes the scenario where C connects his or her TD to an insecure home PC which relays data between TD and M. If the protocol is finished successfully, C inserts his/her bank card into TD and authorises the payment by entering a PIN or a password after verifying the order information displayed on TD (denoted as step 6). An e-cheque is then generated and sent to M (denoted as step 7). The dashed line denotes sending via an empirical channel. We recommend using *https* or telephony as the empirical channel between M and C in this case.

[21]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 [90].

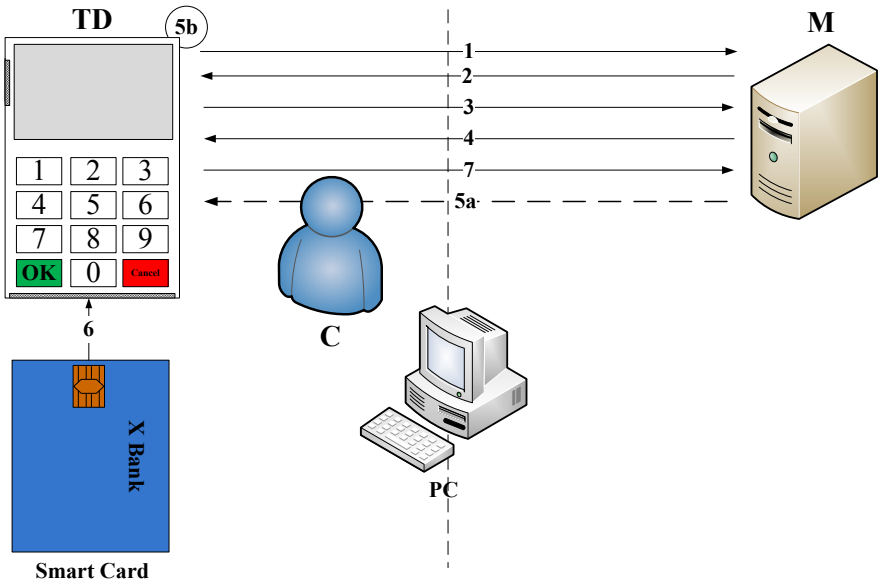


Figure 4.14: Using a HISP (demonstration of a successful run).

To remove the user's complacency in step (5a), we force the user to type the digits of the received digest value into their mobile phone. Similarly, we require the user to input the digest value manually on TD. Therefore the minimum value of D is 6, plus the effort of entering the 4-digit PIN; the total value of D is 10.

To enhance security we may utilise necessary context information during the digest comparison process. For example, we may display the payee's photo, location, company/shop logo and other available context information that helps the payer to identify the payee. When entering the digest value, the payer can be more certain that he/she is paying whom or what he/she wants to pay. We therefore call the use of context information of the payee as secondary security.

4.10 Implementation

4.10.1 System structure

To create a system that can demonstrate the functions of TD, we need one PC acting as the customer's home PC and a second PC acting as M. We also need a program running as a simple data switch on the customer's PC, which applies no cryptography functions and simply relays data between TD and M.

4.10.2 Implementation of TD

In implementing TD, we are concerned about its security and cost. There is little conflict between these goals, since we believe that the limited power and functionality of TD offers reduced opportunities for insecurity; and the simpler it is, the lower the cost. More specifically, we hope that the number of chips in this device that deal with credit card information is as small as possible and the circuit design as simple as possible, because the micro-communication within the device may or may not be encrypted. The latter gives a chance of exploitation, whereas the former increases the cost of computation. In addition, a highly integrated solution will reduce the time of development, and probably reduce the total cost as well. Therefore an integrated design is desirable for making secure and cheap TD.

The function of TD can be divided simply into three parts: reading information, processing information and outputting information. These functions will be associated with the input module, the microcontroller and the output module respectively. Here the input and output modules will not be discussed because these modules – for example, the display screen, the key pad or the USB connection interface – depend on the specific implementation requirement and cannot be further integrated. And those modules can be very cheap; for example, a typical USB phone with display for internet call can be bought for around 30 US dollars [16]. Therefore only the microcontroller is to be discussed in this chapter. An overview of the device design is shown in Figure 4.15.

To achieve this, the microcontroller should be integrated with communication interfaces that read smart cards and support communication between TD and PC, so that we do not need to introduce extra chips and design extra circuits. Secondly, it should be capable of significant computation. From the analysis of the pair-wise TD protocol, we need to perform one public key encryption operation, three hash operations, one symmetric key encryption and decryption operation; and the cost of computing public key encryption is much higher than other cryptography operations. Microcontrollers commonly used cannot satisfy this requirement because of their limited memory resources and computation power.

Therefore, finding a microcontroller that fulfills the above requirement is essential to prove the viability of our solution. Fortunately, such microcontrollers have been available for a few years. They are especially built as microcontroller for smart card readers. Compared with an ordinary microcontroller, they are integrated with communication interfaces to support card reader applications; they have relatively large data memories that enable the microcontroller to do more complicated computation; and they are featured with high performance in computation with powerful instructions and high MIPS.

An example is the Atmel AVR 8-bit AT90SCR100 microcontroller for smart card readers. This microcontroller can run up to 16MIPS and has 64Kbytes of Flash memory, 4Kbytes of EEPROM and 4Kbytes of Internal SRAM. Another producer, Teridian, has a series of integrated 8-bit microcontrollers for smart card readers, for

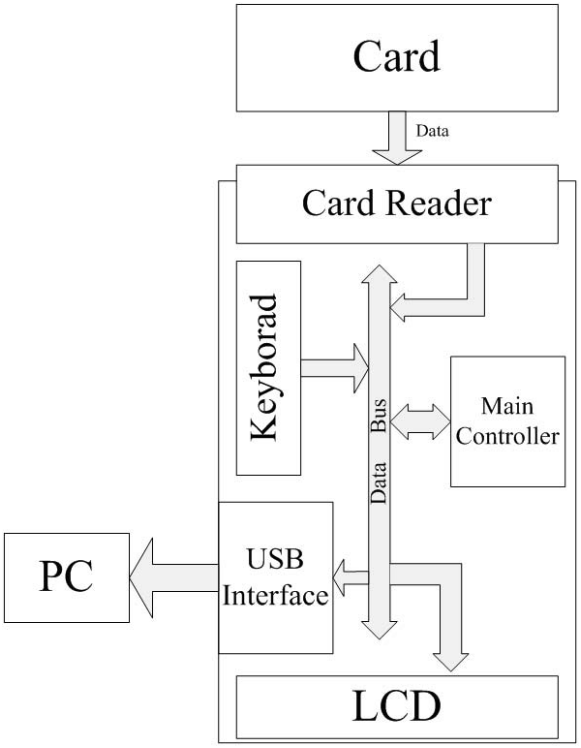


Figure 4.15: A design of the card reader.

example, 73S1217F has a computation power of 24MHz (up to 20 MIPS), together with 64Kbytes Flash memory, 2Kbytes XRAM and 256 bytes IRAM. It is smaller than a five-pence coin and capable of performing public key encryption function. Both of these are available for much less than $10.

More recently, semiconductor producer NXP (previously known as Philips Semiconductors) published a series of powerful secure contact PKI smart card chips; the SmartMX family. An example introduced in [91] shows that a SmartXA2 chip has 7Kbytes RAM, 256 Kbytes of ROM and 144 Kbytes of EEPROM. And its FameXE coprocessor is capable of computing up to 8192-bit RSA public key encryption [21]. Starting in November 2010, more than 60 million new German National Identity Cards using SmartMX chip will be issued in the next 10 years [26]. If banks would adopt such powerful smart card chips in the future, a much simpler and cheaper smart card reader could be produced with the capability of running a HISP.

Our experimental implementation is made on the Teridian 73S1217F evaluation board (see Fig 4.16). The program on the microcontroller is coded in C, and we have interpreted a few core functions into assembly language in order to speed up the public key encryption.

If the card is to remain separated from (and be read by) TD. We recommend making TD read-only and stateless, since this has obvious security advantages.

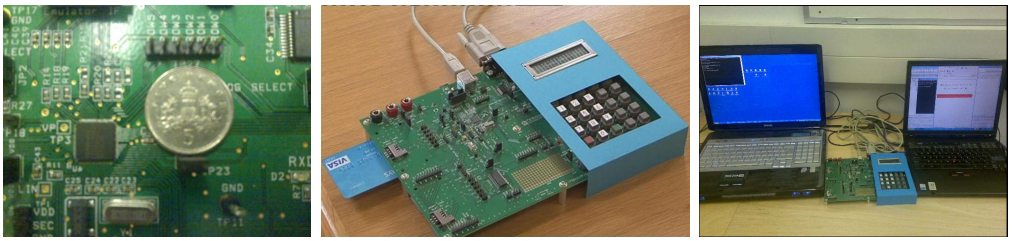


Figure 4.16: TD and the system

4.10.3 Implementation of the software on the PC

TD has to be first connected with a PC which then relays data between TD and the merchant's server. A piece of software is required on the PC to fulfil this task. It should have the following two functions: (1) relaying data between TD and the merchant's server; (2) automatically detecting the payment action. When the customer clicks the payment button, the software should know which merchant's server to connect to.

For demonstration purposes, we have simplified the implementation of the software on the PC: the address of the merchant's server is hardcoded into the software, instead of being automatically read from the web-browser.

4.11 Evaluation

4.11.1 Performance analysis

The time results of computing different cryptography functions on 73S1217 are:

- RSA-1024 with public exponent 65537: 3871 ms for encrypting 80 bytes of data;
- SHA-1: 730 ms for hashing 4Kbytes of data;
- AES-128[22]: 168 ms for encrypting 1Kbytes of data.

We can see that the public key encryption function requires a much longer time to compute than other functions do. To improve its computation, we have already optimised the assembly code[23] of its core operations. Besides programming, a possible direction for further optimisation is to reduce the public key length.

Unlike solutions using pre-installed long term public keys, we consider the public key as one-time only when establishing the connection between TD and the merchant. TD does not record and reuse the public key. Therefore, TD authenticates a potentially fresh and uncertified public key from the merchant in each payment.

[22] We optimised the computation of AES-128 by using look-up tables.
[23] We first compile the C code into assembly by using Keil C compiler.

This provides opportunities of using a shorter key. For example, the valid time of a long-term (one year) public key is 8760 times longer than that of a short-term (one hour) public key. According to NIST standards [38, 130], the accepted minimum length of a long-term RSA public key modulus is 1024-bit, and the minimum public exponent is 65537. If we can prove that a shorter key length, for example, 512-bit, can satisfy the security requirement of a one-time only public key, then a cheaper microcontroller can be used because theoretically, the computation cost of 512-bit RSA public key encryption is only one quarter of a 1024-bit operation.

An alternative solution is to delay the computation of the public key encryption after the transmission of hk_M, which means we could compute $Encrypt_{pk_M}(k)$ during the time of the customer inputting the digest value, of which the time could be long enough. For example, the customer does not only need to input the digest value but also needs to check the order information displayed on TD. A result shows [90] that copying and entering a 6-digit number on the mobile phone takes an average time of 17 seconds.

By carefully designing the payment process and the program on TD, we can use cheaper microcontrollers when necessary.

4.11.2 Security analysis

The security of our solution is obtained from three facts: (A) the use of a HISP and the selection of empirical channels; (B) the use of a dedicated TD; (C) and the use of an e-cheque. Figure 4.17 shows the attack graph of TD. Table 4.14 and Table 4.15 show the evaluation results of TD.

• Credential harvesting

A connected device may require the PC to install drivers, or using the universal plug-and-play function [112], and those protocols of establishing connections between the TDs and the PCs may also increase the risks on both sides [143].

In general, we believe TD should be read-only and stateless in order to isolate itself from any possible malware. This may introduce two challenges:

1. TD will not be able to update its software on its own; therefore, TD will have a limited life which would increase the cost of distribution. One solution is to update the software on TD by the distributors (e.g. banks).

2. Pervasive devices like mobile phones (e.g. a mobile phone version of TD) will require extensive configuration in order to meet this requirement; or, in the worst case, they would never achieve this level of security.

However, if we practice in the opposite way by allowing writing on TD, security risks would become inevitable. A balance between cost and benefit needs to be judged by the future implementers.

In our demonstration implementation, TD is designed to be read only, in this way attacks of stealing credentials from TD become inefficient and difficult.

Passive attacks, such as forging web-pages, are prevented because of the use of the trusted display on the TD: genuine information of the merchant will be displayed on the screen and the customer will check this information before authorising the payment. In addition, because important information is encrypted using the bank's public key, fake web-sites cannot reuse the information if the attack is successful.

• Man-in-the-middle

In our example, we recommend two options of empirical channels: (A) *https*; (B) telephony.

We have identified potential attacks in both channels. However, the use of a HISP means it is not necessarily dependent on a single empirical channel. For example, in case of a security breach on the SSL session or the web-browser, the system administrator or the user can use telephony to send the digest value. Although there are risks of using telephony, the flexibility of choosing different empirical channels increases the stability of the system: the likelihood of all three attacks (attacks on *https*, telephony and web-browser) being successful at the same time is lower. We therefore use half-block hatchings on the attack graph to indicate this improvement.

• Man-in-the-shop

Because of the use of an e-cheque, the merchant cannot hold any of the customer's information that can be reused in other payments. Therefore, MITS attacks can be defeated.

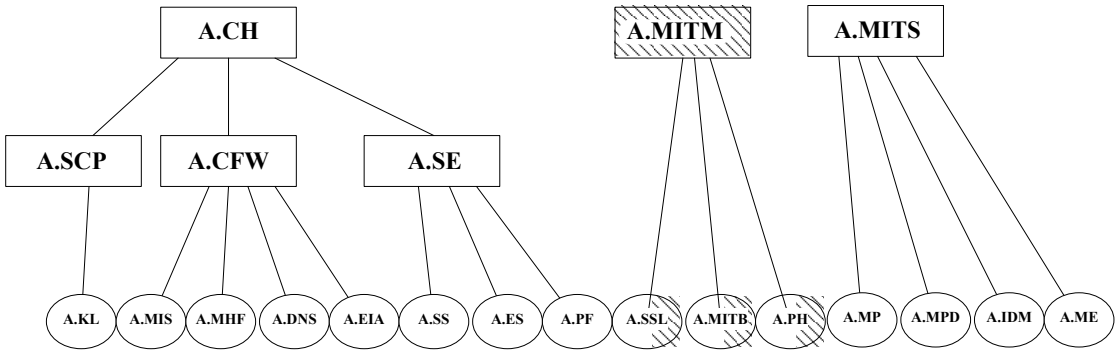


Figure 4.17: The attack graph of TD.

Security	Attack	Likelihood	Impact	Risk
	A.SSL	0.1	50	5
	A.MITB	0.5	50	25
			Overall risk:	30
Complexity	D	M	P	
	10	4	1	

Table 4.14: The evaluation results of TD when using *https*.

Security	Attack	Likelihood	Impact	Risk
	A.PH	0.1	50	5
			Overall risk:	5
Complexity	D	M	P	
	10	4	1	

Table 4.15: The evaluation results of TD when using telephony.

4.11.3 Comparative analysis

Assuming we use telephony to transfer the digest value to HCBK TD, the overall risk of HCBK TD is 5. Figure 4.12 shows the values of overall risks of all eight online payment solutions. The improvement in security is obvious. Comparing with CAP, the main security improvement of HCBK TD comes from the use of phone factor. This is a significant advantage in situations where PKI or SSL is compromised; for example, the customer can request to use telephony to deliver the digest value when he/she is using another person's PC; or the bank enforces the use of telephony in emergencies when the server of the CA of the bank is compromised.

If we choose to use *https* web-page, HCBK TD and CAP share a similar security level. However, Figure 4.19 shows there is a significant improvement in usability compared with CAP. Thanks to the use of the secure connection, a larger amount of payment information can be transferred electronically.

Note there is an implicit improvement in security: when using CAP, the customer has to input details of his/her card onto the browser, which is clearly subject to credential harvesting attacks; successful attackers can reuse these details where CAP is not required. Now we can completely eliminate inputs on the PC by using HCBK TD since the card can provide all the information required in the payment and we can transfer this securely to the merchant by using an e-cheque. The merchant cannot reuse the received e-cheque since it is one-time only and the information about the customer's card is encrypted using the bank's key.

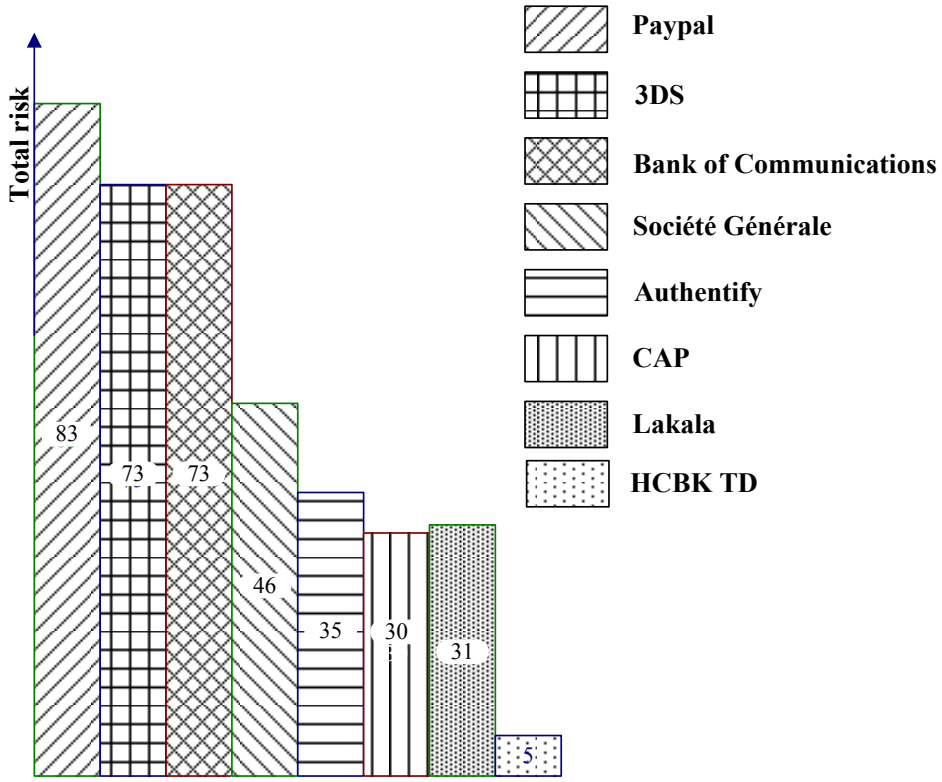


Figure 4.18: A comparison of total risks of eight online payment solutions.

4.12 Related research

The authors of [87, 86] indicate that it is difficult to achieve strong security on PCs for online banking. They present a design of implementing an inexpensive banking dongle with a display. It can communicate with a PC via USB, Bluetooth or by reading a 2D barcode displayed on the PC. This appears to be similar to our design. However, their solution only supports online banking. Since their security is based on the assumption of an existing shared secret, it is inconvenient to be adapted for more pervasive payments such as customer-to-merchant payments. In comparison our solution is more flexible since we can use it for customer-to-merchant payments as well as banking services.

The author of [86] presents an extensive investigation of threats against electronic banking. He produces an attack tree which shows a clear picture of many possible attack techniques which mainly fall in the category of credential harvesting defined in this chapter. In comparison, our evaluation is more systematic and we include and

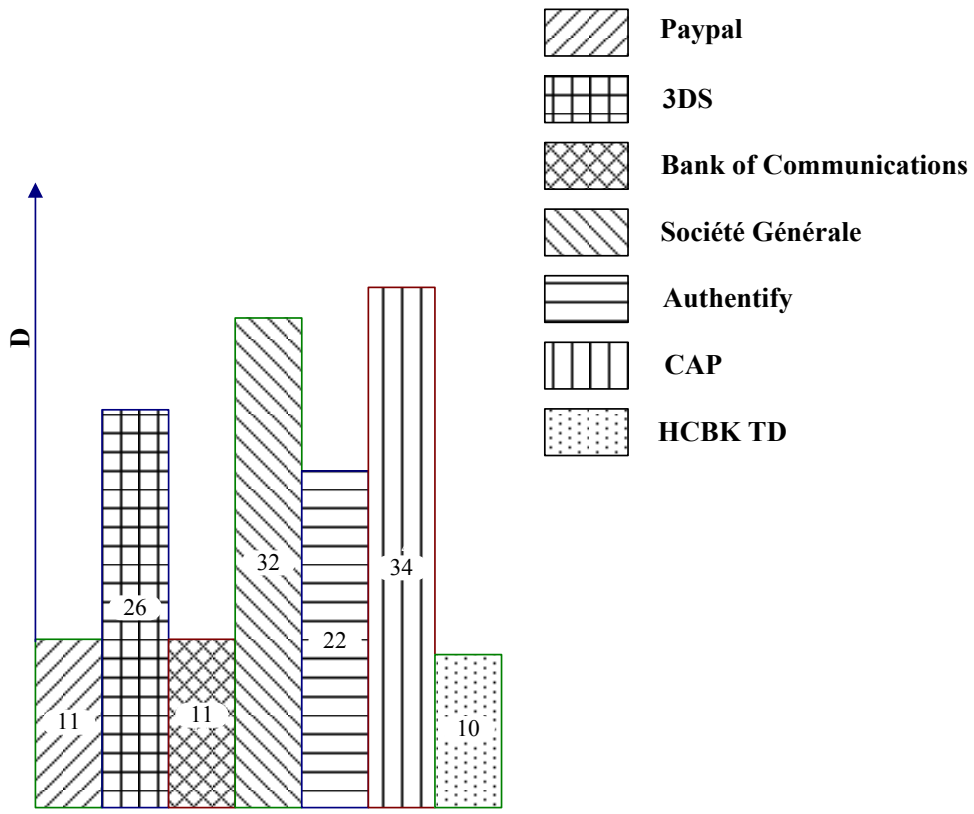


Figure 4.19: A comparison of complexities of seven online payment solutions.

evaluate sophisticated attacks such as MITM and MITS attacks. Our investigation covers a broader area including web-based card payments, e-wallet payment services, online banking and other card payment solutions. Our qualitative analysis allows us to compare different online payment solutions and to identify the most urgent security challenges. We also evaluate and compare the usability of these solutions. In addition, our implementation is more practical which is made on an inexpensive MCU while the implementation in [86] is a simulation made on PCs.

IBM Zurich Lab has developed a card reader called ZTIC [159] with a USB connection to the PC. It acts as a dedicated stand alone "banking web-browser" of a certain bank. By creating a local TLS session, it simply uses the PC as an untrusted data connection between the ZTIC device and the bank server. It is a highly dedicated device with a hard-coded banking web-site address and it requires a pre-configured banking web-site. The amount of data to be processed during an *https* session is large, therefore the computation cost of ZTIC is high. For example, in their tests,

the amount of data to be processed ranges from 30 KB to 600 KB. UBS has adopted this device as its online banking solution.

Some designs suggest using mobile devices as TD. The authors of [127] introduce an anti-phishing solution for online banking, which is made by installing a shared secret received from banks onto mobile phones. Another design is using mobile phones to bootstrap a one-time password out of a long term password for online banking services, which is achieved by using a pre-installed public key from the bank [103].

4.13 Conclusions

Our analysis shows that the role of PCs in transactions must be minimised. Specifically, they must play no role in transactions other than displaying a browser connected to an *https* site known to be associated with the merchant. Furthermore, because of the danger of key-sniffing, it is preferable that no security information is sent from customer to merchant via this route. We note that almost all existing methods fail this last strict test.

Our case study reveals that it is difficult to defend against sophisticated attacks without using dedicated devices. A dedicated payment device can efficiently separate payment information from PCs and it is one device where security is in the hand of the customer. This leads to our research of implementing a TD.

We demonstrate that a HISP is important in improving not only the security but also the usability of TD. This is concluded as Advantage 0:

Advantage 0 A HISP can allow the user to store/access payment account details, card details or other credentials locally on his/her own device; it can allow the user to download the order information from the merchant (the payee) automatically; and the user can securely transmit this sensitive information electronically. The secure electronic connection allows a larger amount of information to be exchanged between the payee and the payer, thereby allowing enhanced and more secure payments.

Considering the expense of using TD, we recommend that TD is more suitable to be used in macropayments where strong security is required. The use of empirical channels in payments allows customers to participate in the process of bootstrapping security and make their own judgements on behalf of their own risks. Our solution has demonstrated one way of efficiently and securely making use of empirical channels as well as the concept of reverse authentication.

One challenge is that we have to produce and distribute TDs in large quantities. The cost can be reduced thanks to the development of integrated card reader MCUs. A possible way of eliminating this cost is to use pervasive mobile devices like mobile phones. In the next chapter we will evaluate different mobile payment solutions, and, later on, we will introduce our new mobile payment solution.

Chapter 5

Mobile payment: case study

5.1 Introduction

This chapter introduces and discusses a few selected mobile payment solutions. The risk evaluation method we use is the same as the one used in the previous chapter. By evaluating different mobile payment solutions, we will reveal differences between online payment and mobile payment. Results obtained in this chapter will be used to produce the requirements of a new design in the next chapter.

Mobile payments are electronic payments made from mobile phones directly to the payee. A mobile phone can be defined as: "any mobile device that contains a smart card that is controlled by a mobile network provider" [40]. Mobile phones are widely considered as convenient tools to support payments because:

- They are pervasive. A report from International Telecommunication Union (ITU) in early 2010 predicted that there would be 5 billion mobile phone subscribers by the end of 2010 [25]. This number is much larger than the number of personal computers (1,026 million in 2010) predicted by ITU [27].

- They are easy to carry and use. This is probably the most important feature of all: mobile phones are designed to be convenient for people to put in their pockets and use in any place at any time.

The scenarios for future mobile payments are highly diversified. The payment solutions that come to dominate the market should be convenient and flexible among different platforms.

Current mobile phones can be divided into two types: feature phones and smart phones. Feature phones are often only suitable to make voice calls and send and receive text messages. The most common mobile payment solution supported by feature phones is text-message based mobile payment. Recently we have seen solutions of using NFC tags or NFC smart cards to "upgrade" feature phones to enable them to support mobile payment functions.

Smart phones are often more powerful in terms of computation, connectivity and sensibility than feature phones. For example, the HTC One X has a 1.5 GHz Quad-core CPU and 1GB RAM; it supports 11 types of connections; it is equipped with 3 types of sensors. They are, therefore, a more convenient platform to support mobile payments. And many mobile payment solutions are only supported on smart phones. We therefore focus on discussing mobile payments on smart phones in this chapter. Unless otherwise stated, the term mobile phone in this chapter means smart phone.

Like PCs, mobile phones today are getting more powerful. The development of hardware and software brings mobile phones more capabilities which can be used to drive the development of mobile payments. One important development among them is the integration of different kinds of Electronic Identities (E-Identities) into phones. It helps reduce the number of cards and tokens a person usually carries; 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 [146]. The FeliCa contactless chip transforms mobile phones into carriers of various forms of identity: transportation card, personal ID card and bank card.

It is reported that in 2012, banks and mobile network operators in the Netherlands will launch a national NFC service, which will enable users to use their mobile phones as payment cards, tickets, coupons or membership cards [24].

In 2010, Chinese mobile network 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 network operators are working together to create a unified national platform for NFC based mobile payment service [23].

In general, we may consider a mobile phone as a bank/payment card once it has logged onto a banking website or an e-wallet website 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 between individuals and 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 it does, in fact, belong to another. You may not know in advance the name of the party to whom you are trying to connect.

To securely transmit an E-Identity, we first need to ensure authenticity as well as integrity of the E-Identity; for example, that 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 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.

In order to find a solution that can solve these challenges, we will evaluate different mobile payment solutions in the market. Before we continue our discussion, we first give an introduction of mobile OSs and mobile phones in the next section.

5.2 Mobile OSs and mobile phones

The functionalities of mobile payment solutions closely depend on the type of mobile OSs and mobile phones and where they are installed. For example, Paypal's Android application requires Android OS v2.1 or higher, while the new Bump payment service of Paypal is only available on iPhone with iOS v4.0 or higher.

According to a report [3] released in November 2011, the most popular four mobile OSs are: Android, Symbian, iOS and RIM OS, and the most popular eight mobile phone producers are: Nokia, Samsung, LG, Apple, ZTE, RIM, HTC and Motorola. In general, the newer the mobile OS or the mobile phone, the more functionality it supports. For example, Android v4.0 supports a face recognition function; iOS v5.0 supports voice control of most applications on the mobile phone; Samsung Galaxy, Google Nexus S and Blackberry Curve[1] have NFC connection; and iPhone 4s has Bluetooth 4.0 connection.

We will not examine details of different mobile OSs and mobile phones in this thesis. However, it is useful to have necessary background knowledge of the current mobile phone market in order to understand the development of mobile payment solutions. Current mobile payment solutions exploit new functionalities on mobile phones; for example, Paypal's iPhone Application uses location data and synchronisation signals generated via motion sensors to identify the relationship between the payer and the payee; Google Wallet mobile application supports NFC based mobile payments; and Alipay[2] mobile application makes payments by taking pictures of 2D barcodes.

In the future, the development of mobile payment solutions will continue to take advantage of the growing varieties of functionalities on mobile phones. However, these new functionalities may expose new vulnerabilities which can be exploited by attackers. It is necessary to identify these potential risks and to find effective solutions in order to improve mobile payment security. In the following section we will establish a risk evaluation model. It will be used to evaluate a few selected examples of mobile payment solutions.

Steps in SP 800-30	Sections
Step 1. System characterisation	Section 5.4
Step 2. Threat Identification	Section 5.5
Step 3. Vulnerability Identification	Section 5.5
Step 4. Control Analysis	Section 5.5
Step 5. Likelihood Determination	Section 5.6
Step 6. Impact Analysis	Section 5.6
Step 7. Risk Determination	Section 5.7
Step 8. Control Recommendations	Section 5.7, 5.9, and Section 6.1 in Chapter 6
Step 9. Results Documentation	Section 5.4 – 5.9, and Section 6.1 in Chapter 6

Table 5.1: The organisation of sections.

5.3 Risk evaluation

SP 800-30 is adopted to evaluate risks in this chapter. Table 5.1 shows the arrangements of sections according to the evaluation process documented in SP 800-30.

5.4 System Characterisation

There are many similarities between online payment and mobile payment. For example, mobile phones today can be used to perform most functions we see on a PC. It is suggested in a report published by NIST [85] that we may consider the threat profile for mobile devices as a superset of the profile for PCs. We may, although not accurate, assume that the threat profile of mobile payment systems is a superset of the profile of online payment systems. This assumption allows us to reuse the material we have presented in the previous chapter.

Figure 5.1 on page 91 shows an example mobile payment system. Our analysis is based on this example. Compared to the system presented in the previous chapter, we can see the differences are minor. We replace the PC with a mobile phone in the customer payment system and we add a mobile phone and a till machine in the merchant payment system. This indicates that by using mobile phones customers can make payments in more places. For example, customers can make payments to people (peer-to-peer payment) who have a mobile phone in the street; we can pay a till machine, an auto-vending machine or an online shop by using a mobile phone. The devices and machines customers can pay does not limit to those displayed in the merchant payment system in Figure 5.1.

These differences, although they seem to be minor in Figure 5.1, have a big impact on security. We observe there are mainly two differences between mobile payment and online payment: payment hardware and software, and payment behaviour.

[1]Blackberry Curve 9350, 9360 and 9370

[2]Alipay is the largest e-wallet payment company in China.

Before we continue our discussion, we make the following two assumptions: (i) mobile payment services become (or will become) pervasive and (ii) users adopt (or will adopt) mobile payment methods to make payment. The two assumptions are important because we analyse potential attacks against mobile payments, and some attacks may only emerge when (i) and (ii) are both true.

5.4.1 Payment hardware and software

Hardware and software used to support mobile payment is different from those used to support online payment. Firstly, the user interface (UI) is often limited on a mobile phone compared to that on a PC. For example, customers need to type on a small phone keypad or a touch screen; information (e.g. web-pages or messages) has to be adapted to be displayed on a small phone screen. Designers, therefore, have to reduce the amount of inputs as well as the information required to complete a payment on mobile phones. This leads to the development of many novel mobile payment applications. For example, some use the phone camera to read 2D barcodes displayed by the payee; some allow the payer and the payee to bump their phones to complete a payment; and some allow users to use their locations to identify whom to pay.

Secondly, mobile phones are equipped with various connectivities, for example, 3G, WiFi, Bluetooth and NFC. These are quickly exploited in many mobile payment services. For example, NFC based mobile payment services.

Thirdly, the software environment on mobile payments is different. This may require us to understand the possible vulnerabilities of different mobile OSs.

5.4.2 Payment behaviour

To understand mobile payment behaviour, we begin our discussion with the following three questions:

1. Where to pay? Mobile payment can be made in many places. Typical examples are payment in the street, payment in a shop and payment at home. Clearly traditional online payment methods cannot cover payment in the street.

2. Whom to pay? In online payment, the merchant is often represented by a shop or a company, for example, a clothes shop, a supermarket, or an online shopping web-site. We often categorize this as customer-to-merchant payment or customer-to-business payment. It can be more accurately described as payment between a personal account and a business account. In mobile payment, the merchant in the scenario of payment in the street is often represented by a person. People usually term this as customer-to-customer payment or peer-to-peer payment. It can be more accurately described as payment between two personal accounts. We can, however, use online payment methods to pay a person. To

more accurately identify this difference, we say the frequency of peer-to-peer payment in mobile payments is higher than that in online payments.

3. When to pay? In mobile payment the payment action is more random than that in online payment. For example, customers may frequently use mobile phones to make payment. One argument that we discovered during an interview is that mobile payment is considered as an efficient tool to support micropayment. Micropayment typically represents payments with values less than 10 pounds[3]. It is pervasive and random in our daily life; for example, to buy a bottle of water, or to buy a bus/tube/movie ticket. However, we can see that online payment also supports micropayment. To more accurately identify this difference, we say that the frequency of a user making mobile payments is higher than he/she making online payments.

Clearly these differences indicate that mobile payment can be highly dynamic: it can be made to anyone, in any place and at any time. It is necessary that we investigate what kind of vulnerability these dynamics may expose and understand their related attacks.

5.5 Establishing the attack model

We have discussed that we may consider the threat profile of mobile phones as a superset of the profile of PCs. This does not indicate that mobile phones are less secure than PCs. A simple comparison of PC security and mobile phone security can be done by measuring the number of known malware. According to a report [117] published by Kaspersky in March 2012, there are 1,590,861 different malicious and potentially unwanted programs detected on user computers in 2011. In another report [104] published by the same company in February 2012, there are in total 6356 pieces of mobile malicious programs recorded as of 1 January 2012. By comparing the two numbers we may conclude that mobile phones appear to be more secure than PCs.

However, we have seen an increasing growth of mobile phone viruses. For example, according to [104] the number of mobile malware has almost quadrupled in 2011. And most mobile malware were discovered on the Android Platform [104]. In comparison, iOS and the RIM system have the least amount of malware being discovered.

In addition, according to our observation of Difference 1, mobile payment applications may use different connectivities, which may expose new vulnerabilities. For example, the recent deployment of NFC mobile payment infrastructure may become targets to MITM attackers. We will, therefore, not only analyse mobile malware attacks, but also evaluate attacks against different connections that can be used to support mobile payments.

[3]There is no uniform definition on the maximum amount of micropayment.

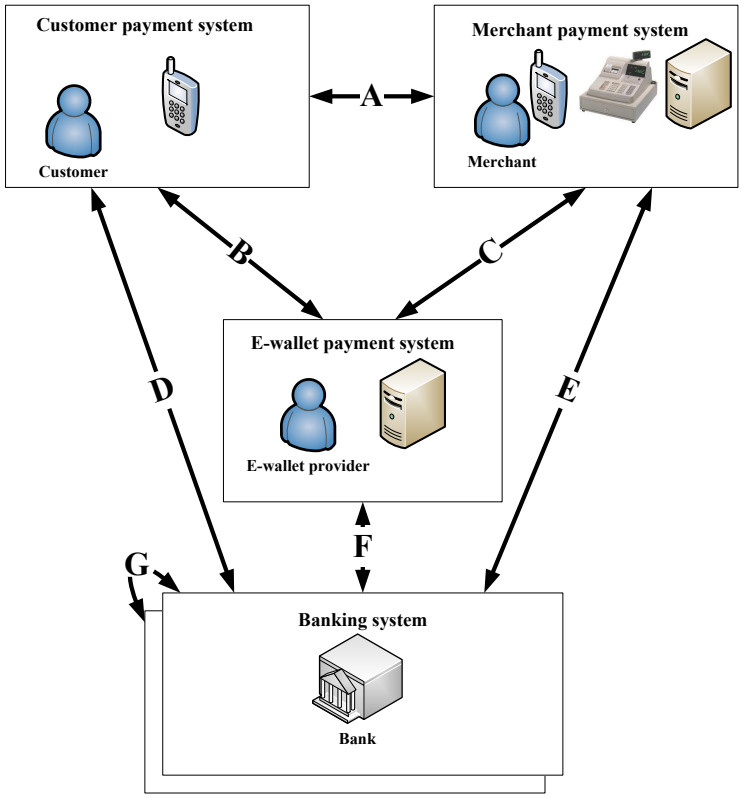


Figure 5.1: An example of the mobile payment system structure.

In Difference 1 we have also indicated that many mobile applications tried to reduce user inputs by incorporating different context into the application, for example, locations. However, context information may not be accurate and they can be forged as well. We will, therefore, identify this as a potential source of attacks in this section.

Difference 2 mainly suggests that mobile payments can be highly dynamic. This can be exploited to develop new attacks against mobile payments. Some example attacks are discussed as following:

1. Attacks that compromising payment infrastructures. When payment is made in places where payment infrastructures are not correctly or closely monitored or maintained, attackers can modify and compromise payment infrastructures to commit attacks. For example, the NFC payment infrastructure on a parking meter in a street can become an easy target to attackers.

2. Attacks from malicious people. We have discussed this as one of the MITS attacks in the previous chapter. However, in mobile payments, the possibility of handing out sensitive information to strangers is much higher. For example,

when making a peer-to-peer mobile payment by using a mobile card reader[4] (installed on the payee's mobile phone), the payer needs to hand out his/her bank card to the payee who then inserts the card into his/her card reader. This provides an opportunity for the payee to copy information on the card; and by compromising the payment software (the mobile payment application) or the hardware (the mobile card reader) he/she can clone the card or commit more advanced attacks like MITM attacks.

3. Attacks that exploiting complacent users. We have indicated that the frequency of making mobile payment can be high. This may increase the possibility of that users become complacent and ignore necessary security processes or notices. For example, users may not correctly and thoroughly check the information sent from the merchants. Attackers may exploit this by replacing names and account numbers in the payment information with their own. In addition, the increased complacency of users may compound to other attacks. For example, users may not carefully examine the payment infrastructures before making payments; they may not carefully and closely monitor the merchant's behaviour after handing out their bank cards.

The first two types of attacks have already been identified and categorised as MITS attacks in the previous chapter. We therefore reuse the descriptions and definitions of MITS attacks in the online payment attack model. However, we observe the possibility of MITS attacks can be much higher in mobile payments because of the "in-the-street" payment types, we call them man-in-the-street (MITS) attacks in this chapter. Note that the acronyms are the same.

Human complacency can be exploited in many attacks. We therefore do not treat this as a single type of attack but consider it as an important fact used in our analysis.

In this section, we will establish a new attack model on the basis of the online payment attack model. Three general types of attacks are selected: credential harvesting, man-in-the-middle and man-in-the-street. Figure 5.2 shows the attack graph.

5.5.1 Credential harvesting (A.CH)

The authors of [64] identified 46 pieces of mobile malware. 28 pieces are used to steal user information, and 24 pieces are used to make premium SMSs. According to a recent report [155], there is a 472% increase in malware between July 2011 and Nov 2011 on Android. 55% of these is Spyware, which steals user information, and 44% are SMS Trojans.

Credential harvesting is the major threat from the increasing number of mobile malware. According to this, we add A.SPY (Spyware) as the main attack vector.

[4]See examples of Square and iZettle in Section 5.7.2

5.5.2 Man-in-the-middle (A.MITM)

The metrics of MITM attacks in mobile payments are often associated with different mobile connections used. At present, NFC, SSL, Bluetooth and SMS are the main connections used in mobile payments.

We have discussed that there is a growing use of context in mobile payments. This provides additional convenience for customers, but it also exposes new vulnerabilities for attackers. We therefore add "using fake context" (A.UFC) as the fifth attack vector in this section.

• NFC hacking (A.NFC)

NFC is based on a short range (<10cm) Radio Frequency channel (13.6 MHz), which assumes that the proximity provides sufficient trust of the data transmitted over this channel.

Examples of NFC-based mobile payment can be found in [129, 88]. An NFC enabled mobile device can act as a card or a terminal, and there is also a mode for peer-to-peer communication, therefore it can be used for peer-to-peer payment. It provides the convenience of simply touching our mobile phones to communicate securely.

However, without link-level security, the transmission between two NFC devices may be subject to eavesdropping and data modification [76]. Experiments [74] show NFC communication can be eavesdropped at a distance of up to 1 metre in active communication mode (when the NFC chip has power); and in passive communication mode (when the NFC chip has no power), it can be eavesdropped at a distance of up to 4 metres. Relay attacks have been demonstrated in [73, 66].

• SSL hacking (A.SSL)

Current mobile banking services, or mobile wallet services accessed on mobile web-browsers, are mostly based on SSL. The services provided via mobile web-browsers are similar to those found on PC web-browsers. As they all exploit PKI, in order to prove his/her identity, a payee must purchase a public certificate which can be "recognised" by the web-browser. This is suitable for banks and large online payment companies like Paypal, but not practical for small shops or individual persons.

Techniques of hacking SSL in mobile payments are similar to those found in online payments. But since the mobile payment scenario is more ad hoc, human complacency may be increasingly exploited.

• Bluetooth hacking (A.BT)

Bluetooth is probably the most popular short-range communication technology now available. According to the Bluetooth Special Interest Group (SIG), in 2014 Bluetooth will be found in 70% of all handsets and 83% of all netbooks [14]. There are a few

implementations [53, 133] as well as research [161, 67] 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 [84], and its pairing process generally require a long time which makes it inconvenient to use. [96] discusses MITM attacks against Bluetooth.

However, the new Bluetooth v2.1 introduces a Secure Simple Pairing (SSP) [100] scheme which is designed to solve the security problems and falls into the same class of HISPs that we study in this thesis. But this immediately introduces a legacy problem: a communication between a v2.0 device and a v2.1 device will be eventually ended as a v2.0 communication. Any Bluetooth which may fall short of v2.1 is too insecure to support payment.

• SMS hacking (A.SMS)

Telephony is regarded as a relatively secure communication technology despite some known attacks [114]. The attacks against telephony networks usually require much larger strength in both resources and knowledge, 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 [102]. 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. Existence of such a SMS payment scheme provides a motivation for putting resources into attack.

However, attacks against SMS may become much easier when malware is installed on the mobile phone. For example, we have discussed earlier that the number of mobile phone malware keeps increasing at a fast speed, and various malware has been reported that can hijack SMS communication.

• Using fake context (A.UFC)

There is a growing trend to use context to authenticate or identify the relationship between the payer and the payee. For example, by computing the distances. However, if the context used in the computation is fake, then the result of the computation of relationship is incorrect. An MITM attacker can exploit this vulnerability to insert their own identity into the payment.

5.5.3 Man-in-the-street (A.MITS)

The attack vectors in this section are the same as those introduced as the man-in-the-shop attacks in the previous chapter. However, in mobile payment scenarios, we notice the concept of "shop" where payments may take place is extended to anywhere, for example, in the street. As explained earlier, we therefore call them man-in-the-street attacks in this chapter.

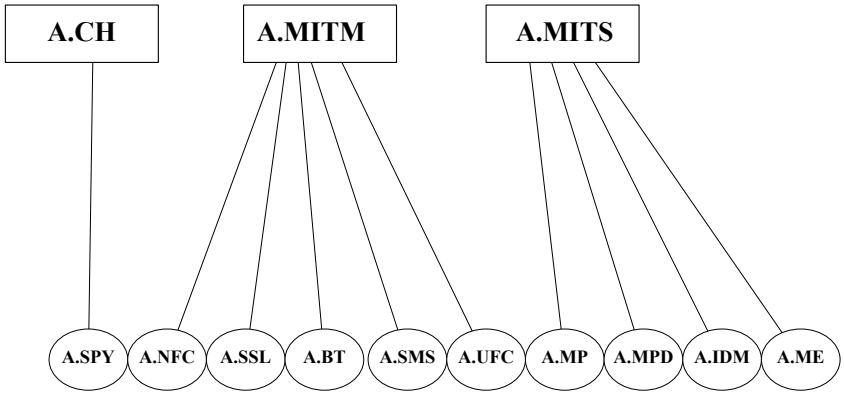


Figure 5.2: The attack graph.

5.6 Quantifying risks

The evaluation model for mobile payments is basically the same as that used for online payments. We therefore only need to present the evaluation guidelines for different attacks in this section. Table 5.2 gives the guideline likelihood values. Table 5.3 gives the guideline impact values.

Table 5.2: The guideline of attack likelihoods.

Attack	Likelihood	Notes
A.SPY	0.5	Spyware is a common mobile phone malware but it is mostly seen on Android and J2ME mobile phone platforms according to the mobile malware statistics [104]. Installing mobile phone anti-virus software can detect this attack. Protection: high. Popularity: medium. Overall: medium.

Continued on next page

Table 5.2 – continued from previous page

Attack	Likelihood	Notes
A.NFC	0.1	Attacks of hacking NFC signal are only seen in experiments and their cost is high. Other attacks are made by using fake context or malicious payment device which are covered in A.UFC and A.MPD. Protection: medium. Popularity: low. Overall: low.
A.SSL	0.5	Exploiting human complacency is the main technique of attacking SSL. Since the frequency is higher and the scenario is more ad hoc in mobile payments, it is more likely to expect complacent customers. Protection: high. Popularity: medium. Overall: medium.
A.BT	0.1	Bluetooth is not widely used in mobile payments. Protection: medium. Popularity: low. Overall: low.
A.SMS	0.5	The cost of direct attack against telephony is high. But mobile malware with capabilities of hijacking SMS is common. Protection: high. Popularity: medium. Overall: medium.
A.UFC	0.1	This attack often requires on-site physical presence of the attacker; for example, being close to the victim, therefore its cost is high. Protection: high. Popularity: low. Overall: low.
A.MP	0.5	Because mobile payments are more ad hoc, they are more likely to involve malicious people. Protection: medium. Popularity: medium. Overall: low.
A.MPD	0.5	Because mobile payments are more ad hoc, it is more likely to involve malicious payment devices. Protection: medium. Popularity: medium. Overall: low.
A.IDM	0.1	The situation with mobile payments is similar to that of online payments. Protection: high. Popularity: low. Overall: low.
A.ME	0.5	Because mobile payments are more ad hoc, they are more likely to be made in a malicious/hostile environment. For example, a customer may have to make payments in someone else's home, in the street or in other unexpected places. Protection: medium. Popularity: medium. Overall: medium.

Table 5.3: The guideline of attack impacts.

Attack	Impact	Notes
A.SPY	50	The impact is limited to the user of the infested mobile phone. Overall impact: medium.
A.NFC	50	The impact is limited to customers within the range of the attacker's signal. Overall impact: medium.
A.SSL	50	The impact is limited to complacent customers. Overall impact: medium.
A.BT	50	The impact is limited to customers who use Bluetooth-based mobile payments. Overal impact: medium.
A.SMS	50	The impact is limited to the user of the infested mobile phone. Overall impact: medium.
A.UFC	50	The impact is limited to a single payment. Overall impact: medium.
A.MP	100	The impact is made to all customers served by the malicious people. Overall impact: high.
A.MPD	100	The impact is made to all customers who use the malicious payment device. Overall impact: high.
A.IDM	100	It may result in the loss of credentials of all customers of the payment service. The impact of this attack is the highest among all impacts of attacks listed in this table. Overall impact: high.
A.ME	100	The impact is made to all customers who are in the malicious environment. Overall impact: high.

5.7 Case study

Mobile payments mainly fall into three categories: peer-to-peer (P2P) money transfer, customer-to-merchant (C2M) payment, and Mobile Banking (accessing bank accounts, withdrawing, depositing, or transferring money between accounts) [59].

The difference between P2P and C2M payments can be small: clearly, we may consider a merchant as an individual person. However, P2P payments present more challenges because of their increased dynamics and mobility. For example, a payment between two strangers in the street is more ad-hoc than a payment to a shop or an online merchant. Normally, P2P and C2M are supported by e-wallet payment companies or mobile network operators.

Mobile banking was first supported by SMS, which is mainly used to deliver notifications of payments being made from banks to customers, or to send requests for

account or transaction information from customers to banks. Wireless Application Protocol (WAP) was later introduced and used to better support mobile banking. Today, most mobile banking is directly based on HTML because of the increase of mobile network bandwidth and mobile phone computing power.

We select a few distinct mobile payment solutions to identify their weaknesses as well as advantages in the following sections.

5.7.1 Peer-to-peer mobile payment

P2P mobile payment requires instant money transfer between personal accounts. However, we discover that the current banking system often does not support this service unless the money transfer is made to a merchant account. Current P2P mobile payments are often achieved via e-wallet payment companies, for example, Paypal[5] and Alipay[6].

• Paypal

The mobile payment application of Paypal takes advantage of the rich context information on mobile phones. For example, Bump[7] is a technology adopted by Paypal to allow convenient mobile payments between two strangers who know little information about each other. Two users "bump" their phones to "find" each other, enabling them to exchange information automatically[8]; for example, the payee can send his/her Paypal account and a request for payment to the payer via a "bump", and the payer can then authorise this payment. This technology is especially useful when the payer and the payee do not know each other and their relationship is identified by context: the synchronised action, the time, and the location. By measuring this context information, the Paypal server can identify their relationship and is able, therefore, to form an electronic connection between the two via its server.

However, although it is possible to prove its convenience properly, it is difficult to prove its security formally. The three factors – the action, the time and the location – together cannot eliminate the attacker's presence; for example, an attacker can stay close to the payer or the payee and bump his/her phone at the same time. The company Bump claims that they can prevent this attack by asking the user to bump again until a uniquely identifiable relationship is found. Given the inaccuracy of current GPS sensors on mobile phones, we found this claim to be fallible. We can prove this by using the following example: providing the defined effective distance of a bump is L, when two users are making a payment, the server will check if there is only one bump within the distance L near each of the user; now if the inaccuracy of

[5] `www.paypal.com`
[6] `www.alipay.com`
[7] `https://bu.mp/`
[8] [15] shows the video footage of this service.

the GPS sensor is $\geq L$, a bump initiated by an attacker can have the chance to pass the check and the attack can, therefore, be successful.

A more advanced attack can be designed as follows: the attacker can tamper with a mobile phone's hardware or software to manipulate the GPS sensor and the accelerometer (the motion sensor); this allows the attacker to carry out automatic world-wide attacks on payments using Bump.

Nonetheless, this application is the first payment application we know of that utilises both the physical touch (the motion) and the location to complete payments. It demonstrates that context has been increasingly used in payments where usability can be improved by removing names.

However, the above analysis has shown that accurate context information may not be available on mobile phones. Context information needs to be authenticated before using it in payment applications.

The security analysis of this application is as follows:

1. Credential harvesting. The mobile application is subject to Spyware attacks if the mobile phone where the application is installed is compromised.

2. MITM. The application uses standard http/https connection and the certificate is preloaded in the application. Therefore, it is immune from SSL attacks. According to the analysis above, this application is subject to context fraud attack.

3. MITS. Because the customer stores his/her bank account and card details on Paypal's server, it is subject to possible data loss on the server side.

Figure 5.3 shows the attack graph of Paypal. Table 5.4 shows its evaluation results.

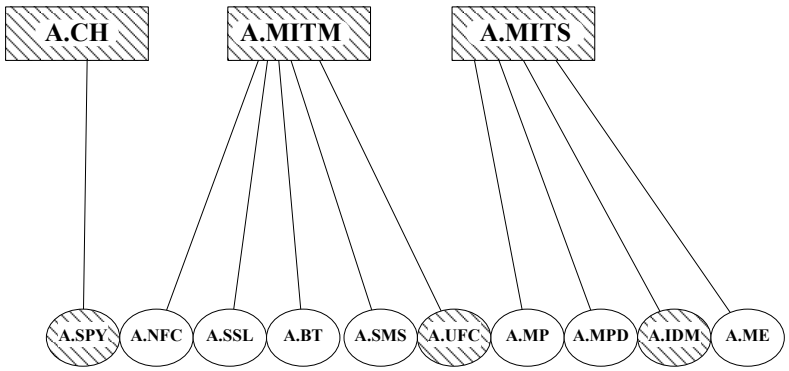


Figure 5.3: The attack graph of Paypal mobile application.

Attack	Likelihood	Impact	Risk
A.SPY	0.5	50	25
A.UFC	0.1	50	5
A.IDM	0.1	100	10
		Overall risk:	40

Table 5.4: The security evaluation results of Paypal.

• Alipay

Alipay mobile payment application provides an extensive usage of 1D/2D barcodes; for example, the customer can scan a barcode to search online for the corresponding product; when making a payment, the customer can generate a 2D barcode, a one-time password to identify the customer's account, which is to be scanned by the merchant. This technique, by scanning the 2D barcode, is largely used to improve usability: the user no longer needs to manually input large amount of data. The security is, however, limited because the visual channel is one way from the customer to the merchant, the customer needs to have assurance as well. For example, the customer scans a 2D barcode from the merchant. See Figure 5.4 for the attack graph. Table 5.5 shows the evaluation results of Alipay.

We note that both Paypal's Bump service and Alipay's 2D barcode payment allows only collocated payments where the payer and the payee is close to each other. This is limited by the context information used in these applications. An interesting direction for future development is to allow convenient mobile payments at a longer distance.

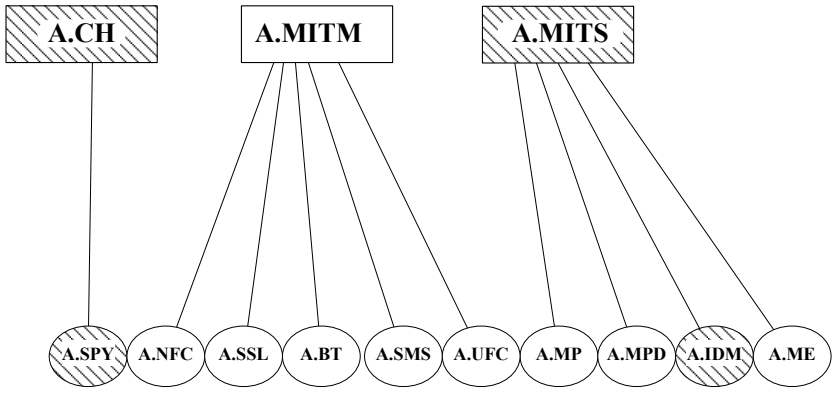


Figure 5.4: The attack graph of Alipay mobile application.

Attack	Likelihood	Impact	Risk
A.SPY	0.5	50	25
A.IDM	0.1	100	10
		Overall risk:	35

Table 5.5: The security evaluation results of Alipay mobile application.

5.7.2 Customer-to-merchant mobile payment

The development of customer-to-merchant mobile payment is mainstream in the current mobile payment market. For example, NFC mobile payments, mobile wallet provided by e-wallet payment companies, and new payment methods based on card readers.

• NFC mobile payment

Currently the most common use of NFC in payments is bus and underground tube traffic cards. Using NFC mobile payment means mobile phones needs to have NFC hardware installed. According to our investigation there are three solutions to install the NFC hardware component on mobile phones:

1. Releasing new mobile phones. New phones with NFC functions are being released in large scale in 2011; for example, Samsung Galaxy, Blackberry Curve[9] and new Nokia phones[10]. A list of NFC enabled mobile phones can be found in [6].

2. Attaching NFC tags/stickers to the mobile phone shells. For example, iPhone users in Japan can use NFC stickers to upgrade their mobile phones for NFC payments [8].

3. Using NFC-enabled SIM cards. For example, it is reported that China Mobile is releasing SIM-based NFC cards on an extremely large scale [101].

Using proximity as the only authenticator can lead to attacks. For example, an NFC relay attack on mobile phones is demonstrated in [66]. In addition, a lack of proper security protocols may make the NFC mobile payment an easier target to MITM attackers [30]. We therefore conclude that proximity alone does not provide sufficient security.

There has similarly been recent publicity regarding attacks on car keys based on near-field radio [152]: one attacker stands next to the car owner whose key is in his pocket, and the other opens and drives away the car. They use long-range radio to transmit the cryptographic challenges and responses between the car and the key,

[9]Blackberry Curve 9350, 9360 and 9370.
[10]Nokia 600, 700, and 701.

through communicating with each of them using short-range radio. We might call this a "two men in the middle" attack! It is desirable that NFC based communication needs to be enhanced by introducing a security protocol that addresses the MITM attack [30]. 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.

One of the most distinct advantages of NFC mobile payment is its obvious convenience: the payment can be made by a single touch. For example, Google Wallet requires only one touch to complete an NFC mobile payment. However, according to the attack model, proximity does not eliminate the possibilities of MITM attacks. In addition, a single touch does not make the payment transparent. We can therefore identify two potential risks of NFC payments:

1. The user may not know what payment he/she is authorising.

2. The user may not know who takes the money.

The attacker can exploit this non-transparency by tampering with the NFC touch pad and creating fake payments.

We discovered[11] that some NFC mobile payment solutions display order information on the mobile phone, and the user can check it before authorising the payment. For example, the user will be asked to touch twice to make the payment: the first touch receives the order information from the merchant, then the user checks the transaction information; the second touch sends back the authorising information to the merchant. This solution calls for manual verification explicitly.

An interesting argument we learned in a meeting with a Chinese bank is that the user will receive a text message including a receipt for the NFC payment that has been made; they argued that in this way the payment can be secured even when the NFC connection is compromised: the user will check the text message and knows exactly what has been payed for. Compared to the previous solution, this one calls for manual verification implicitly: the verification comes after the payment has been made.

However, the two solutions may all fail because they are vulnerable to human complacency. For example, the customer may say "yes" without checking properly, especially when making NFC mobile payment is a frequent day-to-day action. The second solution has an additional logic flaw: the attack detection should be made before authorising the payment rather than after.

Because of these risks, NFC is commonly only used for micropayments, and of course it is limited to short-range mobile payments.

In the future we may see more NFC mobile payment implementations. Some examples are: (A) NFC mobile payment to parking meters [54]; (B) NFC mobile payment to posters [7].

[11]This case was learned during our meeting with China Mobile which introduced its new NFC mobile payment solution. This solution is not seen in the market at the moment.

The two examples show the convenience of pervasive payments in the future: we can buy goods at any time in any place conveniently using our mobile phones. However, it raises our concern that these payment infrastructures can be easily tampered with in pervasive environments. For example, the unattended NFC parking meter can be replaced or tampered with by the attacker to make different payments. And it is even easier for the attacker to replace or tamper with an NFC poster.

This indicates that anyone making a general-purpose NFC payment should have to confirm details of the payment. This is not as necessary where the details are confirmed by context, as with a special purpose card on Tube.

Figure 5.5 shows the attack graph for NFC mobile payments. Table 5.6 shows its evaluation results.

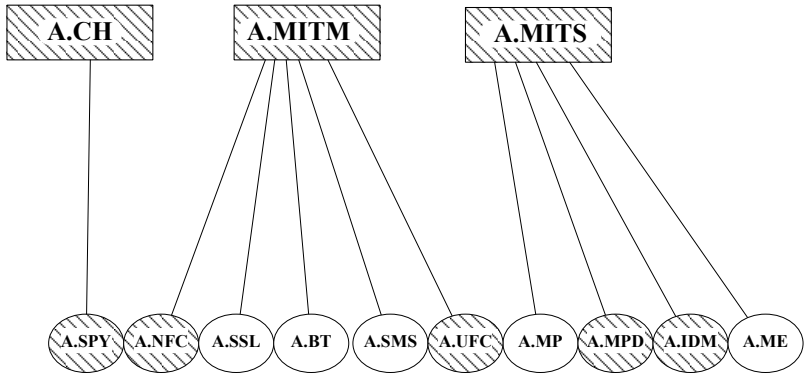


Figure 5.5: The attack graph of NFC mobile payments.

Attack	Likelihood	Impact	Risk
A.SPY	0.5	50	25
A.NFC	0.1	50	5
A.UFC	0.1	50	5
A.MPD	0.5	100	50
A.IDM	0.1	100	10
		Overall risk:	95

Table 5.6: The security evaluation results of NFC mobile payments.

• Mobile wallets provided by mobile network operators

One of the most distinct advantage of this type of solutions is that the customer can immediately convert his/her mobile phone account into a payment account. For

example, one needs to establish an account to pay for his/her phone bills; and this account can also be used to make other payments if its mobile operator has such a service. Fig 5.6 is a representation of the web-based payment model of China Mobile:

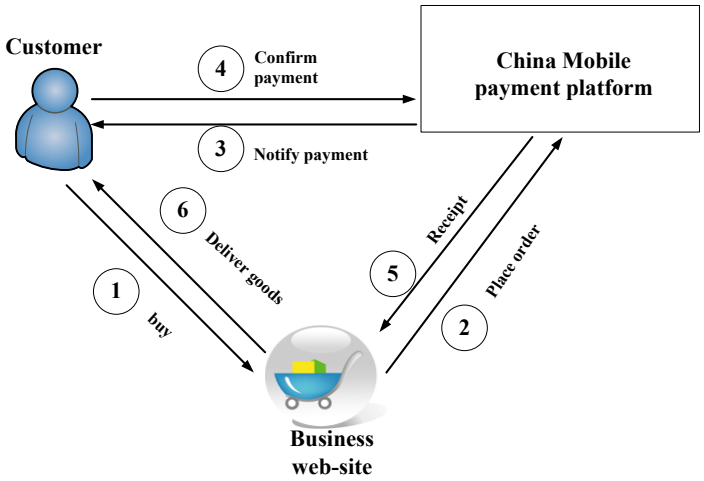


Figure 5.6: The mobile payment processes of China Mobile.

We can see this looks similar to the Paypal online payment example introduced in the previous chapter. A customer can directly use his/her mobile phone account as an e-wallet to make payment.

A slightly different method is to use SMS to make payment. For example, in Fig 5.6 Step 4 is replaced by sending back an SMS from the customer to China Mobile. Step 3 is replaced by displaying a notice on the merchant's web-site. The SMS contains the Order ID: a 9-digit number. The Order ID is displayed on the merchant's web-site. The SMS is sent to China Mobile SMS Payment Service[12]. The customer will receive a receipt when the payment is made.

We focus on evaluating the SMS payment since a similar example of web-based payment has been discussed earlier. The SMS payment method uses an empirical channel to transfer the Order ID from the merchant's web-site to the customer's mobile phone: the customer reads the number from the web-page and types it into the phone. This authenticates that the customer is paying the correct instance of the merchant.

However, the Order ID gives no information of what the customer is paying for. A more robust way is to input the merchant's name, the merchant's ID, the Order ID, the payment amount, and the date and time into the SMS. This, however, requires too much human effort.

[12]In our test, this number is 1065800885560.

An attacker can exploit this by using the following techniques:

1. The attacker can set up a fraudulent website M' disguised as web-site M and direct the customer to M'. The customer inputs the Order ID displayed on M'. This payment is then made to M'. The loss of this payment can reach to the maximum amount allowed in a single payment by China Mobile.

2. The attacker has installed a piece of malware on the customer's mobile phone to hijack the SMS communication. The malware then sends SMSs to China Mobile according to the attacker's commands. The loss of this attack can reach the maximum amount allowed in a period of time (until the malware is removed) by China Mobile.

3. The attacker has installed a piece of malware on the customer's PC. The malware is then used to carry out MITB attacks which can defeat *https*. The attacker can manipulate the Order ID displayed on the web-page. The loss of this attack depends on the number of SMS payments made on that PC.

To make the situation worse, we have discovered some web-sites (with SMS payment services) do not use *https* when displaying the Order ID. This leads to further vulnerabilities. Note that we do not identify A.SPY in this case since passive attacks cannot obtain any useful information in this case. Figure 5.7 shows its attack graph. Table 5.7 shows its evaluation results.

One solution to mitigate these attacks is to use a secure electronic connection to transfer a large set of information from the merchant to the customer. The customer can check the order information before sending out the confirmation SMS.

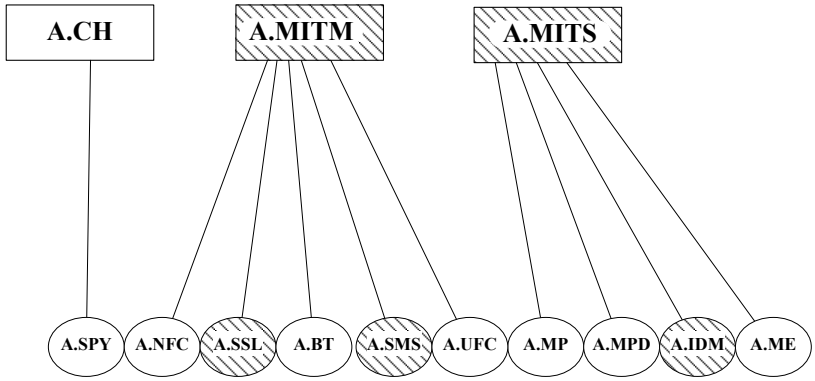


Figure 5.7: The attack graph of China Mobile's remote payment method.

Attack	Likelihood	Impact	Risk
A.SMS	0.5	50	25
A.SSL	0.5	50	25
A.IDM	0.1	100	10
		Overall risk:	60

Table 5.7: The security evaluation results of China Mobile's remote payment method.

• Card reader based mobile payment

Personal mobile card readers have been developed in early 2000, however these are often too expensive to be used widely. Only very recently do we see a growing development in card readers on mobile phones; for example, Square[13] provides card readers for swipe cards on iPhones and Android; iZettle[14] provides card readers for chip-and-PIN cards on iPhones. These card readers have to be plugged into the mobile phone, and then the phone and the card reader will act as a payment device that can take card payments. It takes advantage of the huge user base: card-based payment is already a prevailing payment method.

A general process of making payment via these card readers is as following: the merchant downloads and configures the application; the merchant inserts the card reader into his/her mobile phone; the customer hands his/her credit card to the merchant; the merchant enters the payment amount and inserts the card into the card reader; the customer writes down his/her signature on the phone screen as a symbol[15] to authorise the payment.

This business model was once described as "Payment on the go" by iZettle in their recent advertisement [28] where they emphasise that anyone in any kind of role can take card payments. This, however, raises our concern: handing out our bank cards to strangers is a bad practice. Serious risks can be found here, as random individuals can now use their own devices to take card payments.

According to the mobile payment attack model, we can evaluate the following risks with card reader based mobile payment:

1. People. In order to make a payment, customers are "forced" to hand their cards to strangers. Strangers who take our cards can then have the opportunity to read and remember a customer's card details. For example, one can easily read and remember the security digits printed on the back of our cards when he/she can have it in his/her hands; or, in the worst case, one may simply take the card and run away.

2. Payment device. Copying card details was once made on special devices[16].

[13] `https://squareup.com/`

[14] `https://www.izettle.com/`

[15] The customer's signature has no effect on the payment immediately. It is used for auditing purpose and will be included in the electronic receipt.

[16] See [94] for examples of card skimming.

Usually this attack technique has to be made together with social engineering: the attacker must hide the skimmer from the customer. Now Square and iZettle have made this attack easier: the attacker can tamper with the card reader or produce a fake card reader; the customer will not be disturbed when the attacker is cloning the card in front of him/her.

3. Data. The customer's card data are read and processed on the stranger's mobile phone. An attacker can create a fake application that can record this data. For example, except for the card data, an attacker can easily copy the customer's signature.

4. Environments. The customer (the payer) may not know how to use Square or iZettle because there are no requirements for installing the application on the payer's mobile phones. Therefore a stranger can dominate the entire environment. For example, an attacker tells the customer that he/she needs to input his/her PIN in order to make the payment just like payments in shops. More attack techniques can be developed because of the possible lack of knowledge of the customer.

At the Blackhat conference, August 2011, researchers published and demonstrated details of an attack against Square [113]. They demonstrated that the attacker can copy a swipe card and store the data as a digital file, which can be played without the presence of the card. The CEO of Verifone, the world's largest card reader producer, had published similar accusations [135].

They indicate that the vulnerability originates from the hardware: the card data are not encrypted. However, we discover that the vulnerability is rooted in the basic design of the payment process: we should not hand out our cards to others, and we should not use the devices of others.

One argument from Square in defense of their product is this: the customer can check the notification message sent from Square. This may also fail because of human complacency. Figure 5.8 shows the attack graph of these mobile card readers. Table 5.8 shows its evaluation results.

Attack	Likelihood	Impact	Risk
A.SPY	0.5	50	25
A.MP	0.5	100	50
A.MPD	0.5	100	50
A.IDM	0.1	100	10
A.ME	0.5	100	50
		Overall risk:	185

Table 5.8: The security evaluation results of mobile card readers.

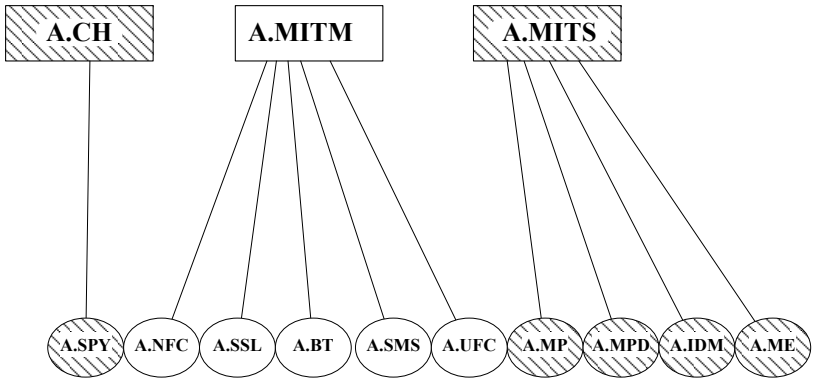


Figure 5.8: The attack graph of mobile card readers.

A possible reason behind these designs is to improve usability by eliminating the input of merchant's details. Because the merchant's application is preconfigured, the customer can simply use the merchant's mobile phone and card reader and do not need to make many inputs.

One solution to mitigate these attacks is to allow the customer to use his/her cards on his/her own mobile phone via his/her own card reader. However, this will cause a usability problem: the customer will have to enter manually all the necessary information of the merchant.

If we can find a way to conveniently and securely transfer information between two devices, then we can solve this problem by allowing the customer to make payments on his/her own devices.

5.7.3 Mobile banking

• SMS based mobile banking

We have tested SMS mobile banking services from Barclays, RBS and ICBC[17]. Barclays and RBS only provide text alert services which send SMSs from the bank to the customer to inform them of details regarding transactions being made and account balances.

ICBC supports a larger function set. For example, except for request for information functions, they allow money transfer via SMSs. The customer can write a command like "HK#your card number#recipient's card/account number#recipient's name#amount#payment password" in an SMS and send it to the bank. Here HK represents money transfer. The phone number for SMS banking is static and it is

[17]ICBC stands for Industrial and Commercial Bank of China. It is one of the largest banks in the world.

published by ICBC on their web-site.

Inputting a large amount of data on the constrained mobile phone user interface can lead to low usability. We believe this is one of the most significant disadvantages of SMS based mobile banking. Figure 5.9 shows the attack graph for SMS based mobile banking. Table 5.9 shows the evaluation results of SMS based mobile banking.

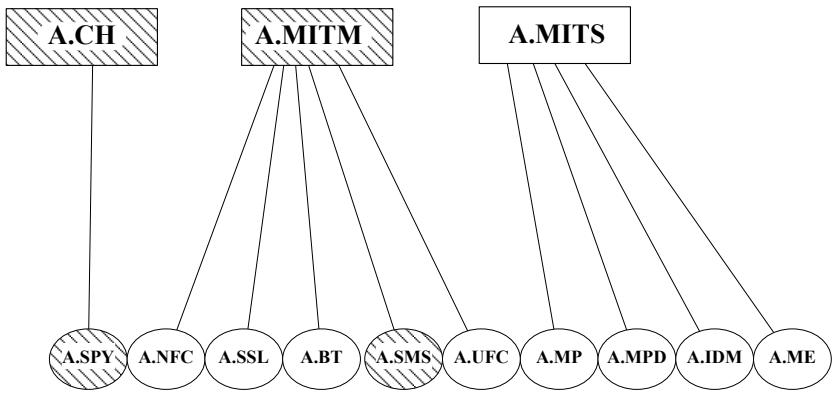


Figure 5.9: The attack graph of SMS based mobile banking.

Attack	Likelihood	Impact	Risk
A.SPY	0.5	50	25
A.SMS	0.5	50	25
		Overall risk:	50

Table 5.9: The security evaluation results of SMS based mobile banking.

• Mobile application based mobile banking

Banks release their own mobile applications providing banking services. Their functionalities are similar to those found in web-based online banking.

The problem of the large number of different mobile banking applications may lie in usability. A customer who has accounts in multiple banks may have to download and install multiple mobile banking applications and not get confused when using different ones. One solution is to consolidate different bank accounts into one application as the payment interfaces of different banks are often similar to each other.

China UnionPay[18] provides a unified payment platform[19] for mobile businesses. It released a software plug-in tool (we will call it UnionPay Plug-in) for mobile applications. This tool is used to support "in-application purchase" service on mobile platforms. The customer can use all Chinese bank accounts on this platform by registering his/her bank accounts or cards. It provides standard payment interfaces which can be used by commercial companies to produce mobile applications that can accept payments from China UnionPay. A unified payment platform has clear benefits in terms of usability:

1. The customer only needs to configure once before use.
2. The customer can manage and use multiple bank accounts and cards on the same platform.
3. The customer only needs to learn to use one payment method.

The UnionPay Plug-in, as described in UnionPay's documents, is designed as a software module to be included in the mobile application released by the merchant. Figure 5.10 shows the payment process as described in one of their documents.

In Figure 5.10 in Step 3 we can see that the payment interface is called within the application. In Step 4 the customer needs to enter a few information to authorise the payment. We list its details as follows:

1. Read the order information and press the confirm button to make a payment.
2. Input its mobile phone number in order to get an authentication code via SMS.
3. Input the authentication code received by SMS together with the 4-character CAPTCHA code.
4. Choose a bank card and enter the card's number and password.
5. Press the confirm button to proceed.

In terms of authenticating the customer, its method is robust because of the use of multi-factor authentication. Note that they require the input of the mobile phone number in every payment. It means they will always check whether the mobile phone number input by the customer matches with the one registered or not.

However, we discover there is a fatal drawback in their solution: the customer may not be able to authenticate the UnionPay Plug-in installed in the merchant's application. For example, a malicious merchant or a malicious programmer can lure the customer to install a mobile application (or a fake one) that has a fake UnionPay

[18]China UnionPay is the largest bankcard association in China.

[19]Our study is based on the confidential documents provided by China UnionPay. The mobile payment platform has not yet been released in the market. Accessed date: Nov. 2011.

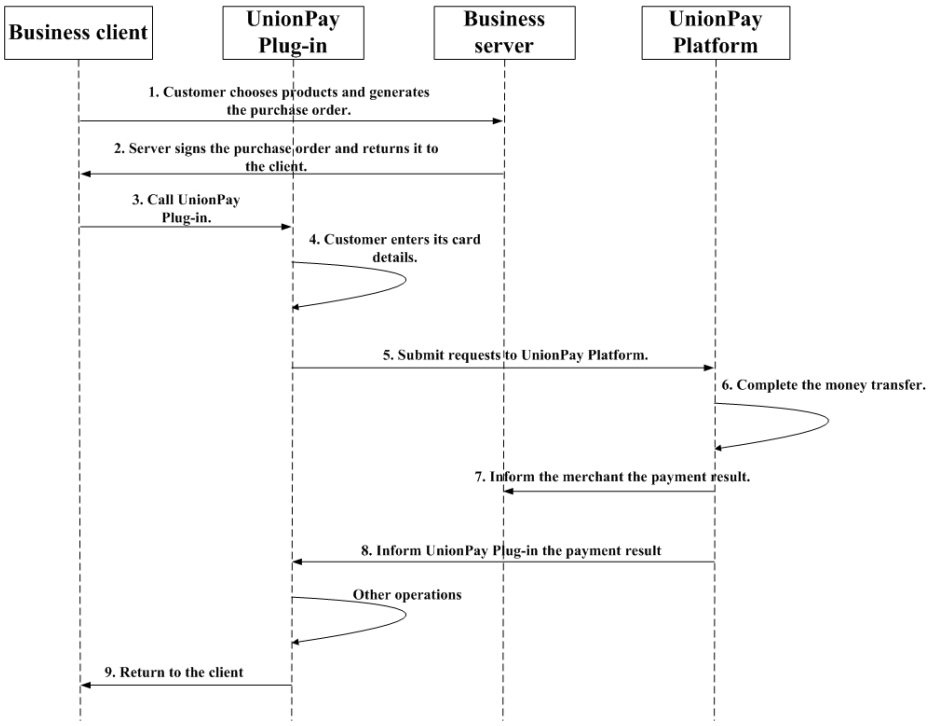


Figure 5.10: The payment process using China UnionPay Plug-in

Plug-in which has exactly the same interface as an authentic UnionPay Plug-in does. In this way, the merchant or the attacker can manipulate the payment in real-time. For example, it can steal information from the customer or commit MITM attacks.

In addition, allowing developers and companies to implement their own mobile applications by including UnionPay Plug-in may lead to the following security flaws:

1. It is difficult to control the quality of the mobile application. Low quality mobile applications may expose vulnerabilities that can be exploited by attackers.

2. The number of mobile applications that have UnionPay Plug-in integrated can be large. It is difficult to ensure that all customers will download these mobile applications from legitimate sources. An attacker may implement a malicious application and trick the customer into downloading and installing it.

We therefore identify A.SPY, A.SSL, A.SMS, A.IDM and A.ME as potential attacks against UnionPay Plug-in. Figure 5.11 shows the attack graph of the mobile payment platform of China UnionPay. Table 5.10 shows its evaluation results.

To combat these attacks, we suggest that UnionPay Plug-in should be used as an independent mobile payment application, and we can then use this application as

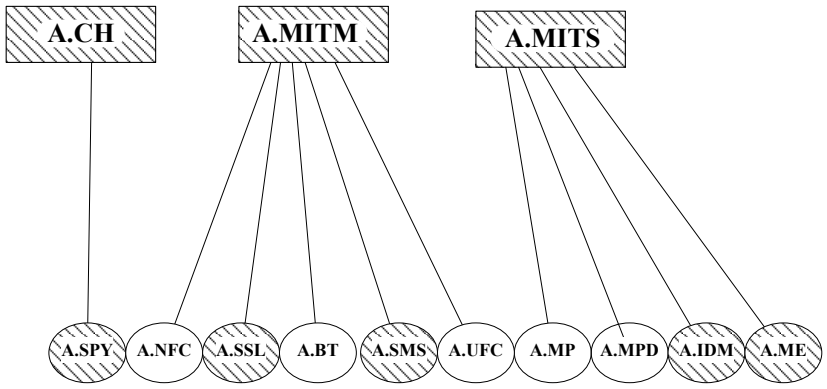


Figure 5.11: The attack graph of the mobile payment platform of China UnionPay.

Attack	Likelihood	Impact	Risk
A.SPY	0.5	50	25
A.SSL	0.5	50	25
A.SMS	0.5	50	25
A.IDM	0.1	100	10
A.ME	0.5	100	50
		Overall risk:	135

Table 5.10: The security evaluation results of the mobile payment platform of China UnionPay.

the entrance to access to different merchants' services. By "reversing" their business process, we can reduce the possibility of using a compromised payment interface.

Another solution is to use UnionPay Plug-in as an independent mobile payment application, and other mobile applications send order information to it. This solution has three major advantages:

1. More secure. We can isolate the payment application from other applications.

2. Easier to implement. Merchants can still implement their own mobile applications and they will only need to use a communication interface to send order information to the UnionPay mobile payment application.

3. Wider usages. The UnionPay mobile payment application cannot only be used to make payment to mobile applications, but to many other entities and services. For example, to pay an online-shopping website displayed on another device (e.g. a PC); to pay a shop or a restaurant; or to pay another person with a mobile phone (peer-to-peer mobile payment). This is clearly a critical attribute to have in order to produce a popular mobile payment application.

However, this solution introduces two difficult challenges: (i) how to conveniently transfer the order information from the merchant to the customer electronically? (ii) how to authenticate the order information received from the merchant? We will continue our discussion of creating a new design to solve the two challenges in the next chapter.

• New types of mobile banking

A German company[20] produced a dongle with a camera to secure online banking. When making a payment, the dongle reads a 2D barcode displayed on the online banking web-page and decodes it into the payee's International Bank Account Number (IBAN), amount of payment, and then generates a 6-digit authentication code, which is to be read and inputted by the customer on the web-page. We may, in a simpler way, treat this 2D barcode as a digital signature, and the dongle as a digital signature reader. If the attacker cannot modify this digital signature, this technology protects against malware attacks on the PC; for example, the MITB attack.

5.8 Discussion: more examples

In this section we introduce some interesting examples that worth mentioning but are not included in our risk evaluation. The Lakala example introduced in Section 4.7.4 shows a new design of distributing unmanned machines in the public to allow users to use their cards to make various online payments. We are recently informed that the same company is releasing a wide range of mobile card readers[21], for example, card readers that can be plugged into mobile phones; card readers (without displays) that can operate by connecting to PCs; card readers (with displays) that can operate by plugging a landline cable; and landline telephones (with displays) with card readers. They all provide similar functions to those of the Lakala machines introduced earlier in the previous chapter. Prices of these range from 20 pounds to 60 pounds.

This shows a possible direction in the future: people can purchase various payment devices to conveniently and securely access to their bank accounts at any time in any place. Their service can be enhanced by providing services of paying persons or merchants.

Barclays released its mobile payment application Pingit in April 2012. It can make person-to-person payment by using mobile phone numbers. It is more convenient than the pay by telephone number service provided by Bank of Communication since Barclays enables this by using a mobile application which does not only enjoy more mobility but also take advantage of the automatic access to the phone contacts.

This is a significant development: banks are now moving to the market of swift person-to-person payments. This will surely introduce a competition between banks

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

[21] See examples on `www.lakala.com`

and e-wallet companies. The argument of a possible improvement of the example of Bank of Communication in the previous chapter can be applied to Barclays as well: "in practice, telephone numbers may not be the only option to replace account details, for example, we can use email addresses, social network accounts, nick names, addresses, domain names or even photos to identify the payee as long as this information is authentic."

These visions support our payment solution design presented in the next chapter.

5.9 Conclusions

The mobile phone platform provides new opportunities for developing mobile payment solutions. These mobile payment solutions clearly provide more convenient payment options for users. However, in pervasive environments, the attacker is likely to have more opportunities of attacking as well. We have discussed that proximity cannot eliminate the possibility of attacks because the attacker can get close to the user in crowded environments; exposing payment APIs to be used by third-party companies and developers may introduce MITS attacks; and using mobile card readers on the mobile phones of others is not secure because it is not safe to trust the devices and software of others.

The hardware and software environment on mobile phones are not safe if they are not used properly; for example, downloading applications from unofficial resources, or downloading applications developed by companies or developers with low credibility; and the security of NFC, Bluetooth and telephony may not be strong enough to defend against sophisticated attacks. The growing variety of mobile payment applications and mobile phones makes the task of protecting mobile payments even more challenging. The complexity of this problem keeps increasing.

This poses a great challenge to security research: how can we protect mobile payments in such diversified, dynamic and hostile environments? We present our solution in the next chapter.

Chapter 6

Mobile payment: building a unified mobile payment platform

6.1 Introduction: requirements of our new design

This chapter presents our design of a secure mobile payment solution. We will first discuss requirements of this design by comparing examples introduced in the previous chapter. We then give the structure of our design which completes three main functions. In order to improve usability, we will reveal a new naming strategy that can be used to facilitate the payment process. Demonstration implementations are also made and evaluations are presented at the end of this chapter.

Figure 6.1 on page 117 shows the overall risks of seven mobile payment solutions we have studied. We observed the following facts:

1. The mobile payment solution that used mobile card readers in Section 5.7.2 was ranked as the most insecure solution considered. It is a mistake to require the customer to use another person's mobile phone and mobile card reader to complete the payment: it is never a good idea to trust others without sufficient knowledge.

2. The use of the NFC connection does not improve security. Proximity alone does not provide sufficient security.

3. Integrating mobile banking interfaces into many mobile applications may expose vulnerabilities that can be exploited by MITS attackers.

4. Third-party authentication may not be sufficient in securing frequent and ad hoc payments. Names only are neither sufficient nor convenient in authenticating the payee in ad hoc scenarios because of human complacency.

5. Using context without proper authentication in payments may expose new vulnerabilities.

6. All mobile payment solutions are subject to mobile malware attacks.

Except for these, additional challenges may appear as the development of mobile payment continues. We assume that mobile phones will replace wallets in the future. Cash, cards and cheques will be digitised and used on mobile phones. We will rely on using mobile phones to make day-to-day payments. We observe this development may introduce the following challenges:

1. Mobile payment may become increasingly popular in peer-to-peer payments and payments to automated machines. For example, the Paypal Bump service facilitates the payment between two persons; the mobile card reader allows card payments between two persons; and the development of NFC mobile payments allows more pervasive payments to machines or other NFC-enabled facilities. Mobile payment solutions need to be flexible and convenient in order to satisfy these requirements.

2. There can be many different mobile payment companies. The customer may have to obtain and learn multiple mobile payment solutions in order to cope with different payment needs. This increases the difficulty of making mobile payments and hence slow down its development. For example, the customer has to configure and manage multiple payment applications; the customer needs to set and remember multiple passwords, which may introduce additional dangers that the customer may use only one password in different applications; the customer also has to submit his/her card details or banking details to multiple payment companies. In addition, security standards of different mobile payment solutions are different. Compromising one may degrade the security of others in this case.

3. The number of attack vectors may increase. Because new mobile payment solutions keep emerging, new vulnerabilities are likely to be found and hence new attacks. It is desirable to develop a security framework that not only satisfies the increasing security requirements, but also can be adapted to allow novel implementations of new payment solutions.

To solve these challenges, we propose a design of a new mobile payment platform that can allow implementations of secure, convenient and flexible mobile payment solutions. We call it the HCBK Mobile Payment Platform. In general, achieving more ubiquity is our main object. Requirements for this are discussed in the following sections.

6.1.1 Reducing MITS attacks

According to the attack model established in the previous chapter, there are four kinds of MITS attacks: A.MP, A.MPD, A.IDM and A.ME. Awareness of these attacks

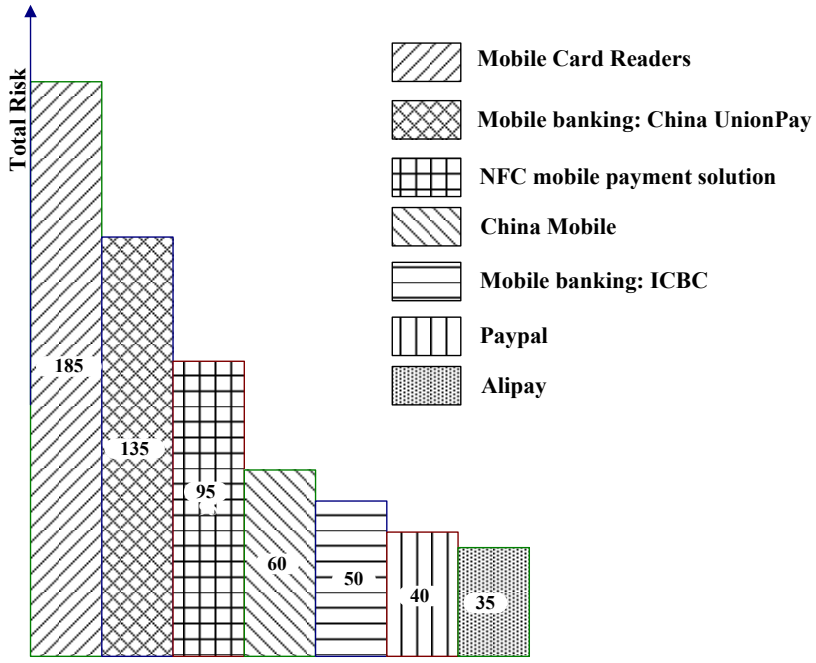


Figure 6.1: Overall risks of seven mobile payment solutions.

should be raised because (A) mobile payments are expected to be carried out in ad hoc places, for example, in streets, small shops, small restaurants and many other places that are difficult to predict; (B) in mobile payments we are more likely to pay someone whom we do not know and hence we should not trust; for example, strangers we meet in the street, small vendors we come across, staff in small shops or restaurants, and other individual service providers; (C) the frequency of people making mobile payments may increase quickly in the future, human complacency may become a serious problem.

The illustration of "Payment on the Go" by iZettle, discussed in the previous chapter, provides a good concept of just how ad hoc mobile payments are expected to be. However, lessons are also learned from the mobile card reader case study that the risk of using mobile phones or devices belonging to others in payments is high.

There is no effective way of defending against A.IDM because it is difficult to enforce any security improvement on the side of merchants or service providers. However, we can avoid using payment services from third parties by directly using mobile banking to make payments.

Keeping secrets in the customers' own hands is a general technique of protection against A.MP, A.MPD and A.ME. We should never hand out our cards or account details to someone we do not trust. For example, when using a mobile card reader, we should use our own mobile phone and mobile card reader to make the payment.

To achieve this, we need to transfer the order information from the payee to the payer so that the payer can complete the payment locally on his/her own device. This requires a secure electronic connection between the payer and the payee.

To establish this connection we need to solve the following challenge: our mobile payment platform should be connection agnostic. In other words, it needs to be capable of using any available connection. We will discuss this in the next section.

6.1.2 Making use of all possible electronic connections

The development of mobile payments is often restricted by hardware; for example, in order to use NFC mobile payments, we need to have a mobile phone with NFC connection and another device with NFC connection to receive payments. It is difficult to require that all customers to have mobile phones with NFC connections in the near future. Therefore, it is important to allow customers to make use of any possible electronic connection available to make payments, for example, WiFi, GPRS, 3G, Bluetooth, NFC, RFID or Infrared.

In order to use such a wide variety of connections uniformly, we must assume that the initial connection is insecure. For example, the attack model introduced in the previous chapter lists known vulnerabilities of the four main connection types. In the future, new types of connections may appear as the development of mobile phone hardware continues. For example, the Android 4.0 Ice Cream Sandwich includes "WiFi Direct" feature [19] which allows the customer to create an ad hoc network by using WiFi easily. Therefore, this assumption is necessary in order to make our mobile payment platform compatible.

The flexibility of the mobile payment platform is improved when the customer is allowed to choose any available connection to use. For example, the customer can use NFC connection when there is an NFC touch-pad to receive the payment; the customer can use WiFi, or cellular data to connect to the Internet to complete the payment; or the customer can use WiFi, Bluetooth or RFID to connect to the merchant's device when there is no Internet. This flexibility is desirable in practice. For example, when making NFC mobile payments, there are situations where the customer feels it is necessary to keep a safe and comfortable distance from the merchant, for instance, women in the Middle East, or when there are obstacles to prevent the customer from getting close to the merchant easily, such as customers sitting in cars.

An important requirement arises when we try to achieve our goal of using all means of electronic connections: how to connect two devices easily? Convenience is no doubt one of the most important factors in succeeding in the mobile payment market. We discuss this in the next section.

6.1.3 Improving usability

If a mobile payment solution is secure but inconvenient to use, it is likely to fail in the current mobile payment market. We discovered this fact during our meetings with

banks and mobile payment companies.

There are continuous efforts to improve usability while improving security in mobile payment; for example, NFC is a typical attempt because touching devices is indeed simple and straightforward; and Paypal's Bump service creates a virtual connection between the payer and the payee by bumping each other's device together. These solutions all try to solve the following challenge: how to securely establish an electronic connection between two devices with minimum cost of human effort? This can often be helped by using context. We will discuss this in the next section.

Another issue in mobile payments is to make the payment process simple. We have seen requirements for manually verifying the order information before or after authorising the payment. These requirements are vague and not easy to fulfil because of human complacency.

6.1.4 Using context with assurance

NFC uses proximity to identify the relationship between the payer and the payee. Paypal's Bump service uses GPS location data and the bump action to compute and identify relationships. Proximities, locations and actions are context information. These can be used to reduce the human effort of manually inputting the device's machine address. However, we have discussed that they are vulnerable to MITM attacks because the attacker can use fake context information. We therefore require that context can only be used with assurance.

6.1.5 Reducing human complacency

The definition of complacency from the *Oxford Dictionary of English* (third edition) is: "a feeling of smug or uncritical satisfaction with oneself or one's achievements". Clearly, we can associate security risks with human complacency.

Human complacency may disable well-designed security. For example, humans frequently fail to choose a robust password [147]. In payments, human complacency can be common because making payments is a frequent day-to-day action. Payment companies or banks often require the customer to check the merchant's information; for example, its account number and name. However, a human may skip this check because he/she is complacent, or he/she may not know the complete set of merchant's information.

We can therefore conclude that a robust payment solution cannot rely on the human to verify the merchant's information unless this is somehow forced.

6.1.6 Reducing the impact of mobile malware

Mobile malware is a serious threat to all mobile payment solutions. A general technique of reducing the impact of mobile malware is to reduce the number of payment

applications to be installed on mobile phones. This improves security as well as usability, as we have indicated earlier.

We suggest that payment interfaces should not be integrated into applications released by merchants. Order information should be transferred explicitly from the merchant to the payment application installed. The customer knows where to manage and make payments, which can reduce the risk of human complacency: the customer only needs to learn to use one piece of software and the number of passwords to remember is reduced; there will be a uniform display of warnings and the customer is more likely to be alerted.

6.2 Platform design

This section gives an overview of our platform design which aims to satisfy the five requirements concluded in the previous section. Three key functions are required to build the mobile payment platform: connecting mobile phones, securing connections, and transferring money.

6.2.1 Connecting mobile devices

Improving convenience is the goal. Our solution is to provide a convenient naming system to use in mobile payments.

In distributed systems, names are the key to allow sharing among devices. This applies in mobile payment as well. Compared with machines, humans use a different naming system which is convenient and flexible so that one can easily name one another, but it is also inaccurate and unstable. For example, human names are traditionally used in local areas and are, therefore, not guaranteed to be globally unique; a human may have different names in different scenarios; and a human may change his/her names.

An additional challenge is that when making a payment, the payer and the payee may have little information about each other; for example, they do not know each other's name. Context is therefore used. For example, in the cash payment scenario, context is both accurate and efficient to guarantee the correctness of a payment.

Machines, on the contrary, use unique and usually long information to identify each other; for example, IP addresses. These machine addresses are not convenient to be used by humans. In addition, on mobile phones, the IP address of each mobile phone is not static.

When making a mobile payment, we have to find a solution to conveniently bridge the relation between two machines and the relation between two humans. We will continue our discussion in Section 6.3.

6.2.2 Securing connections

After the establishment of the electronic connection, secure procedures will be initiated to authenticate and encrypt this connection. Our solution is to use a HISP to authenticate the connection and establish a symmetric key to encrypt the connection. This is similar to the one used in Chapter 4.

6.2.3 Transferring money

After the connection has been secured, the customer will make the payment to the merchant by transferring money to the merchant's account. This can be made in various ways. For example, we have demonstrated that there can be up to three methods to transfer money from the customer to the merchant in the example showed in Figure 5.1. The details of what information is exchanged depends on the payment company or the bank involved in the payment.

Our focus is to find a solution to make the process of transferring money from the customer's account to the merchant's account to be completed within a short time (e.g. a few seconds). For example, in a rushed mobile payment, it is not acceptable to wait for a few minutes for the answer of the money transfer.

During our investigations with banks in the UK and in China, we discovered that instant money transfer between personal accounts is difficult to implement within a bank or between banks. For example, it may take from a few minutes to a few hours to complete a transfer between personal bank accounts; and exceptions of taking a longer time may happen during weekends and bank holidays. This is an obstacle to peer-to-peer mobile payments.

To overcome this obstacle, people often use services from e-wallet companies which support instant money transfer between personal accounts (e.g. between two e-wallet accounts or between one e-wallet account and one bank account that has been registered in the e-wallet company).

The drawbacks of this method are: (A) the customer needs to trust the e-wallet company and sends all his/her bank account details to it; (B) both the customer and the merchant have to use accounts from the same[1] e-wallet company.

In Section 6.5, we will make a proposal that can achieve instant money transfer between personal bank accounts by creating an electronic "contract" between the payer and the payee.

6.3 Naming in mobile payments

In order to find websites or online services, we use Domain Name System (DNS) servers to find their IP addresses. This is true because (A) IP addresses are not

[1] We have not learned any example that people can transfer money between accounts from different e-wallet companies.

easy to remember and (B) websites or online services can sometimes change their IP addresses.

The change of IP addresses is more frequent on personal devices, for example, mobile phones and PCs. This is because personal devices often do not have static IP addresses and they need to obtain a new IP address every time they connect to the Internet. To allow direct online connections between devices, we must establish an online DNS server that provides naming services.

How to establish a DNS server that satisfies our requirement? This question may be interpreted as what kind of name we want to use to address different devices? The following requirements must be satisfied when deciding the name:

1. A user can decide what name to use.

2. A user can have multiple names.

3. A user can frequently change its name.

And we also need to satisfy the following technical requirements:

1. It should be easy to determine the semantics of a name.

2. It should be easy to maintain the uniqueness of a name.

3. It should be easy to allow users to exchange the information of a name.

Traditional naming systems and mechanisms may not solve these challenges efficiently (see discussions of the difficulty of naming in [31]). We propose a new model of naming system called Dynamic DNS (DDNS) to help customers to conveniently establish the electronic connection between devices. Because security will be supplied later on the connection there is no need to make the DDNS server secure.

We call the name used on a DDNS server as Personal Domain Name (PDN). A PDN consists of five types of information. This information will contain much more detail than is apparently needed, but allow parties who only know a fraction of it to identify their intended counterpart. Each type enables unique functions in the naming system:

1. Unique identifiers. The unique identifier is to ensure the PDN is unique within the system. A PDN has at least one unique identifier. Options are: Bluetooth address, WiFi MAC address, email addresses, telephone numbers, IM accounts, social network accounts, online domain names (e.g. OpenID), and unique identifiers generated via registration (a service provided by the implementer). The DDNS server will detect duplicate unique identifiers to ensure each one is unique on the server.

2. User-defined information. The user-defined information is a collection of items that the user wishes to publish in the public. The DDNS server will not check the uniqueness of this information. Options are: unique identifiers, real names and nick-names, addresses, company or organisation names and job titles, description of business services, and other information.

3. Context. This information is sensed by the mobile phone. Contexts, especially locations, are useful to improve the usability of the identification and searching process. Options are: GPS locations, photos, motions, voices, and other context.

4. Auxiliary information. This information is not searchable[2] but can help the users to identify one another. We recommend the use of personal photos or company logos. This information should be displayed together with the search result.

5. Passwords; optionally some applications may be generated by these.

Because we do not ultimately need to trust the DDNS server it is optional whether or not names (including modifications) need to be registered using any security. However whatever degree of security can be provided for this function in a given application will be an advantage since it will make loss of service attacks harder.

The design of PDN allows the user to use any kind of combinations of information to search for one another. Figure 6.2 shows an example of the search process.

In Figure 6.2 an interesting step is to search by context. We have indicated in the previous chapter that obtaining accurate and authentic context information can be difficult. In addition, it is not easy to identify correct payment relationship by measuring context information, for example, by measuring locations.

In our design, we use a HISP to provide the necessary authentication after the establishment of the initial insecure connection. For example, the following logic is an example of identifying payment relationship by measuring locations:

> IF $|Location(A) - Location(B)| \leq L$ AND $Sync(A) = Sync(B)$, THEN A intends to connect to B.
>
> IF A intends to connect to B, THEN connect A to B.
>
> *Verify*(A intends to connect to B). IF False, THEN disconnect A and B.

$Location(A)$ means measuring A's location. $|Location(A) - Location(B)|$ means measuring the distance between A and B. $Sync(A)$ means obtaining synchronisation signals from A. These signals can be generated by time, actions or other context. *Verify*() is a security function achieved by using a HISP. This verification step provides more freedom to the designer when designing functions of computing $|Location(A) -$

[2]This has to be determined by the implementer.

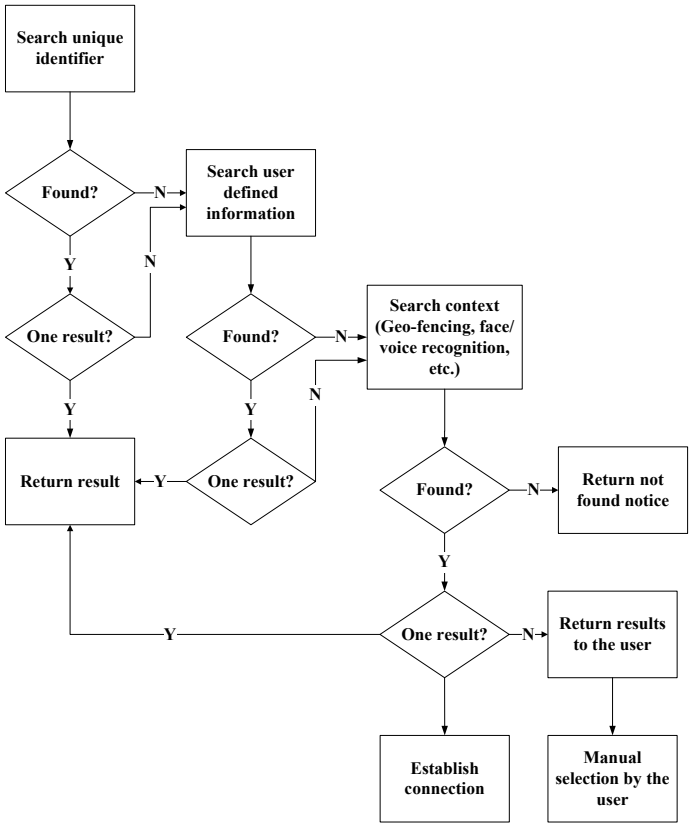


Figure 6.2: An example of the search process.

$Location(B)| \leq L$ and $Sync(A) = Sync(B)$. As long as the last step works, the designer can introduce and use any kind of context to identify the relationship between A and B. This leads to Advantage 3:

Advantage 3 when using context in payments, HISPs can reduce the complexity of the metrics of context to be captured and measured.

To prove this, we use another example of face recognition. Google has released the face recognition API on Android v4.0 in October 2011 and it has developed a new phone-lock function by using face recognition.

Face recognition provides a new way of connecting by context. It can be used to create a new mobile payment application. We give an example of how to construct and use such an application: users upload their photos when registering their payment accounts and they have installed the mobile payment application on their phones.

When user A pays user B, A first takes a photo of B using its mobile phone. A's phone then uses a face-recognition function to search for the matching photo in the payment system's database. If found, the application displays B's profile and the order information (generated on B's device) to A together with a payment dialogue. A enters the amount and press the confirm button to authorise the payment.

However, the face recognition function provided by Google has serious security flaws when using it to identify relationships. We have conducted a test to identify the security flaws in face recognition by using a Samsung Galaxy mobile phone and a HTC Wildfire mobile phone. The author configures the Galaxy mobile phone by taking a photo of himself. The first test shows he can unlock the Galaxy mobile phone by showing his face in front of the camera. In the second test, the author takes a photo of himself using the Wildfire mobile phone and displays this photo in front of the Galaxy mobile phone. It shows the author can unlock the Galaxy mobile phone using the Wildfire mobile phone. Note that the Wildfire mobile phone has a low resolution screen with only 240×320 pixels.

This test shows that the phone-lock function on Galaxy mobile phones using face recognition can be easily defeated by forging the context: showing a photo rather than showing the face. And the consequences of this attack are serious: the attacker can gain access to the resources on the mobile phone. Related reports can be found in [95]. This corresponds to one of our conclusions in Chapter 2: "some contexts can be easily forged".

Like the example of connecting by measuring locations, we can enhance the application of connecting via Face Recognition by adding the following process: *Verify*(A intends to connect to B). IF False, THEN disconnect A and B. Details of how to implement the verification function is presented in the next section.

6.4 Securing the connection by using a HISP

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

1. Customer-to-merchant (phone-to-server): customer C has finished shopping and wants to pay merchant M using his/her mobile phone. This can be an online or a point-of-sale (POS) mobile payment.

2. Peer-to-peer (phone-to-phone): Alice meets Bob in the street. She wants to pay him some money for his painting. She connects her mobile phone to his and makes the payment.

To simplify our discussion, Scenario 1 is discussed in this section, and Scenario 2 is discussed in Section 6.6.

6.4.1 Tailoring a HISP

In our mobile payment scenario, only two parties are involved in the payment: customer and merchant. The following pair-wise protocol is identical to the one used in Chapter 4, except TD becomes D in this case:

1. $D \longrightarrow_N M: ID_D, INFO_D, hash(hk_D, ID_D), hash(k)$
2. $M \longrightarrow_N D: ID_M, INFO_M, pk_M, hash(hk_M, ID_M)$
3. $D \longrightarrow_N M: Encrypt_{pk_M}(k), hk_D$
4. $M \longrightarrow_N D: hk_M$

5a. $M \longrightarrow_E C \longrightarrow_E D: digest(hk_D \oplus hk_M, (ID_D, ID_M, pk_M, k, hash(k), INFO_D, INFO_M))$

5b. D compares the *digest value*[3] with its own version.

6.4.2 The human contribution

Human interactions are not always reliable due to human complacency. To standardise the work flow of using a HISP, we need to clarify steps (5a) and (5b).

In step (5a), when conducting payments at home, the empirical channels C can directly interact with M are phone calls, SMSs, or *https* sessions. When paying in front of a person or a till , the empirical channels can be the visual channel or the voice channel between C and M. We use a dashed line to show the transmission of the digest value in Fig 6.3.

To remove the user's complacency[4], in step (5a), we force the user to type the digits of the received digest value into mobile phone. 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 digest 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 empirical channel, and at the same time the protocol authenticates the uncertified pk_M and establishes the one-time session key k. C is convinced that a secure connection is established between him and M.

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

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

One alternative solution to read digest value is to use mobile phone cameras. M can display a 2D barcode on its device or web-page, and C can use his or her mobile phone to read it. This function may require a high quality mobile phone camera in order to read and decode the 2D barcode quickly and correctly.

6.5 Transferring money

Once the HISP above has been run, there is a channel that the payer trusts as both secret and authentic between the payer's mobile phone and the payee. 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 bootstrapping the session key k by using a HISP. The details of this will largely depend on the actual implementation of banks and merchants.

The session key can now be used to allow secure downloading of order information from M. C 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.

A good candidate is sending an e-cheque, in which C's private information is encrypted under a bank key (and, therefore, not understandable by M), together with all information that is not secret from M. For example, this e-cheque might contain: M's *Order Info* (e.g. M's account details, date and time, amount and other details of the purchase), $Encrypt_{pk_{Bank}}(hash(Order\ Info, C's\ Card\ Details))$. An example protocol is given as below (also see Figure 6.3):

6. $M \longrightarrow_N D$: $Encrypt_k(M's\ Order\ Info)$

7a. C checks *M's Order Info* displayed on D.

7b. If correct, C authorises the payment by entering his/her password on D.

8. $D \longrightarrow_N M$: e-cheque

Note that in Figure 6.3 the representation of M can replaced by other devices as well, for example, a mobile phone, a tablet computer or a till.

E-cheques provide a way of combating sophisticated MITS attacks. M can then forward this e-cheque to a bank to get cash.

In each case the fact that the order information 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.

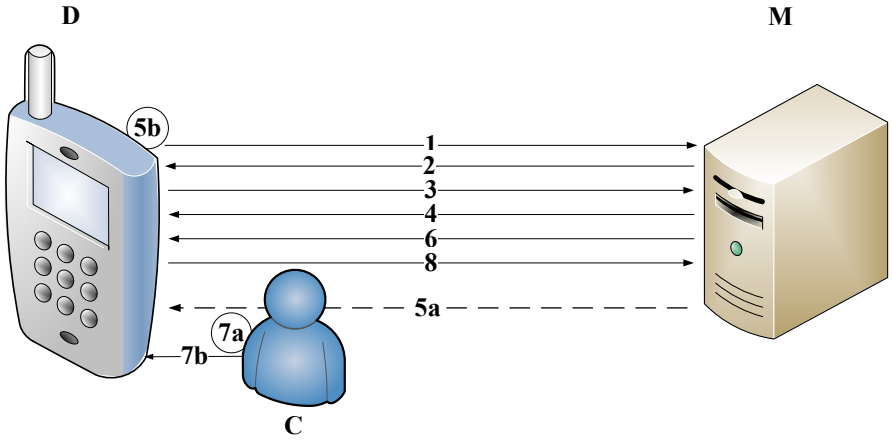


Figure 6.3: Using a HISP (demonstration of a successful run).

The concept of an e-cheque, which can also becomes an e-ticket, e-voucher or e-contract, can be used in many other applications as well.

However, since the process of money transfer is largely determined by banks or payment companies, the use of e-cheque directly between the customer and the merchant may not be available in practice.

One alternative solution is to pay via a bank or a payment company. For example, C can directly send the e-cheque to Bank B, which can decrypt and understand the C's payment request; after verifying the C's account details, Bank B then completes the payment by using the following methods:

1. Transfer money from the C's account to M's account. And then send a signed confirmation to M to inform it that the payment has been made.

2. Freeze the same amount of money described in the order information. Then send a signed confirmation to M to inform it that the payment has been made. And transfer money from C's account to M's account later. This creates a form of "contract" between C and M.

The second method is a possible solution to enable instant money transfer between personal bank accounts. This can be achieved by adapting the existing mobile banking interfaces. If C and M use different banks, C's bank can still send a signed confirmation to M's bank to indicate that C is able to make this payment in the future. M's bank accepts it and send another signed confirmation to M.

6.6 Advanced mobile payment model

Reducing the complexity of connecting two mobile devices (or the mobile device and the server) is important to improve the usability of the entire payment system. We have discussed that a HISP uses two channels: channel N and E. And we have noticed that channel N is usually an insecure high-bandwidth channel. Because there is no need to require channel N to be secure before running the HISP, the initial set up of channel N between the payer and the payee can be made using unauthenticated information about the payee. For example, Bob wants to pay Alice using his own device; Alice's device is identified using Alice's name instead of the IP address; Bob connects to the device using Alice's name. In this example, an attacker may advertise its device using the same name as Alice's. However, a HISP is used to detect the attacker's presence. In other words, the connection between Alice and Bob can be secured later after running a HISP. This leads to Advantage 2:

Advantage 2 When establishing the initial electronic connection in payments, HISPs can facilitate this process by allowing the payer to use unauthenticated information about the payee.

In this section we use the scenario of peer-to-peer payments. We use a server that implements DDNS; we call it the DDNS Server[5]. And both Alice and Bob's mobile phones are connected to this server before the payment via Internet. Alice can find Bob's mobile phone in three ways:

1. Searching unique identifiers. Alice may have Bob's telephone number, email address or social network account stored on her mobile phone beforehand. In this case, Alice can easily select one of these unique identifiers to quickly locate and connect to Bob's mobile phone via the DDNS Server.

2. Searching user-defined information. Alice meets Bob the first time, she knows only Bob's given name (Bob) and Bob's company name or home address; she enters this information and quickly narrows down the returned results using auxiliary information (e.g. photos). She clicks Bob's link returned by the DDNS Server and connects to Bob's mobile phone.

3. Searching context information. Alice and Bob stand close to each other. Alice can connect to Bob's mobile phone by uploading her location data to the DDNS Server. The Server measures locations of the two parties and connects them if their locations are within the defined range.

They can also use short-range connections if they are close to each other. Figure 6.4 shows the advanced mobile payment model. Similarly, we can replace Bob's

[5]The DDNS server we have implemented now supports the following functions: searching unique identifiers (e.g. Bluetooth addresses and email addresses), searching user-defined information (e.g. names and postal addresses), and searching context information (e.g. locations).

phone with a server, a PC, a till or other devices. We can see there are only small differences between this and Figure 6.3. This proves that the use of HISPs gives freedom to the user to use any available connection while the main frame of payments remains the same.

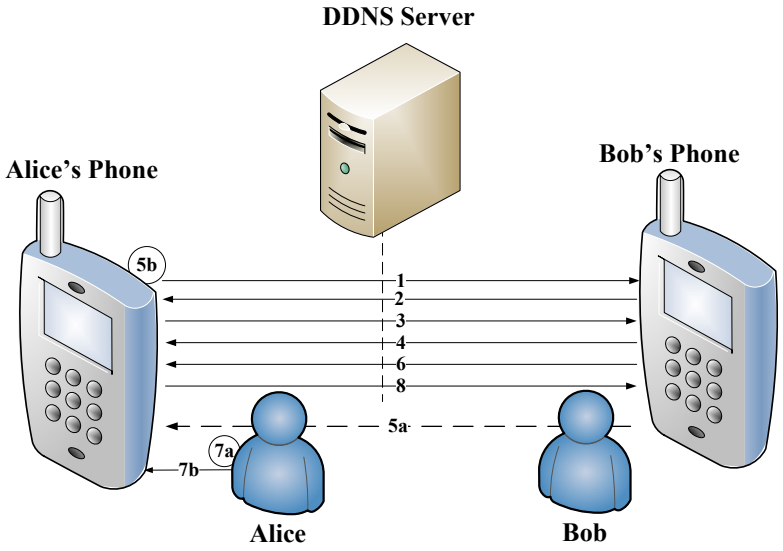


Figure 6.4: Advanced mobile payment model (demonstration of a successful run).

6.7 Demonstration implementations

In demonstration implementations of Scenarios 1 and 2 discussed at the beginning of Section 6.4, we have used the following approaches.

A. A mobile phone is connected to a merchant 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 6.4.

B. Two mobile phones are connected via Internet or Bluetooth: the protocol will start to run after the connection is established. An e-cheque is sent to the payee from the payer.

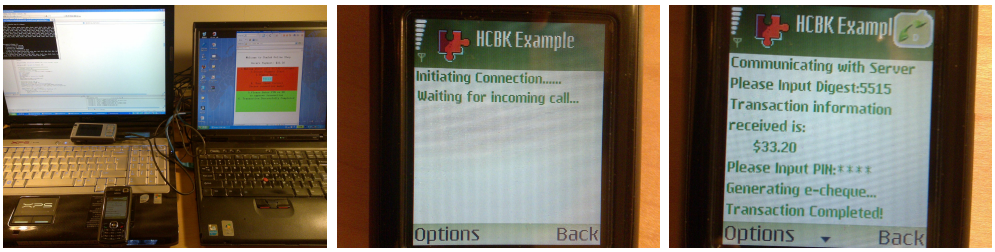


Figure 6.5: Customer-to-merchant mobile payment implementation.

6.7.1 Implementation of approach A

Figure 6.5 shows the snapshots of our implementation of approach A. Its scenario is as follows:

1. The customer C has come to the point of paying during an Internet session and is confident that the *https*[6] session is connected to the merchant M.

2. C presses a button on the website for mobile payment and starts (*) the payment application on his mobile phone. The button gives C's phone payment number to M securely via *https*.

3. M calls C's mobile phone and runs the initial messages of the protocol with it.

4. M calculates the digest and displays it on the existing *https* window.

5. Assuming C wishes to carry on, he types this number into phone which then decides if numbers agree. Agreement gives secure connection.

6. M sends order information over the secure (authenticated and encrypted) connection including amount, name, possible logo, date and time, and bank information.

7. The payment is displayed on the mobile phone and C is asked to confirm payment (*).

8. Payment is processed by e-banking, which generates a "receipt" to send to M.

It will be necessary in practice to have the customer prove his/her identity as part of this process. One or both of the points marked (*) are appropriate. And approach B works similarly.

In this case, a mobile phone acts as a "trusted device". It is required that the user must activate his or her online banking account or any other payment account before

[6] *https* can be replaced by making a phone call, sending a text message, or any other ways that the user trusts. However, it is necessary to understand the strength of these methods. For example, we have pointed out the fallibility of SMS earlier.

or during the payment process. The merchant's order information is downloaded onto the mobile phone. This saves the human effort of inputting the data that required in most current mobile banking applications.

The application does not need to hold the merchant's identity or secrets beforehand because we do not require any configuration before the transaction.

This advantage gives more flexibility when distributing the application to customers. There is no requirement for customers to configure and install any information when dealing with different merchants. However, we do recommend this application to be distributed by banks rather than merchants. This is because: (A) banks are considered more trustworthy than ordinary merchants; and (B) incorporating a set of public keys from major banks can allow customers using e-cheques to improve the security of online transactions.

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 [38, 130].

6.7.2 Implementation of approach B

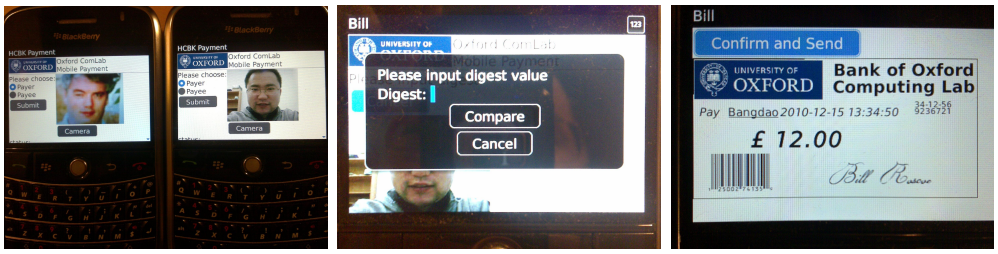


Figure 6.6: Peer-to-peer mobile payment implementation.

Figure 6.6 shows the snapshots of the implementation of approach B. We assume the payer and the payee are close to each other. They can establish an online connection via a DDNS server; or they can use Bluetooth to set up the connection since they are near each other. The photos in Figure 6.6 show images of the users. By incorporating available biometrics or location information (GPS) into the protocol, we can further enhance the security and provide the user with more information to verify each other.

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.

We have implemented a demonstration DDNS server with basic functions to facilitate the creation of the initial insecure online connection. It is also being used

to support some on-going projects in Section 7.4. We will not present details of its implementation in this thesis because they are beyond the scope of our discussion.

6.8 Evaluation

6.8.1 Security analysis

The security analysis is organised according to the attack model introduced in the previous chapter. Figure 6.7 shows its attack graph. Table 6.1, 6.2 and 6.3 show its evaluation results.

• Credential harvesting

Harvesting credentials on mobile phones is a common attack. We have discussed two popular techniques: spyware and SMS trojans. If a mobile phone is compromised by these attacks, our solution alone cannot defend against credential harvesting.

Normally we can detect these attacks by installing mobile anti-virus software; for example, Kaspersky, F-Secure and McAfee. And the risk can be further mitigated by forcing users to download software from the official websites; for example, the iOS on the iPhone requires software to be installed from their official online application shop. 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. More discussions about mobile malware can be found in [64, 40, 98, 65].

• Man-in-the-middle attacks

To carry out a MITM attack, the attacker can (i) hijack the empirical channel or (ii) make a one-shot guess attack[7]. Recall the Empirical Evaluation Criteria presented in Section 1.2.2:

$$1 - PR_C - 1/2^L \geq PR_S \quad \text{AND} \quad \alpha \geq L$$

According to the discussion made in Section 1.2.2, there are different ways to hijack the empirical channel depending on what empirical channel we use.

We recommend three types of empirical channels: face-to-face communication, *https* web-page (based on SSL) and SMS. When the customer and the merchant are

[7] A guess attack means that the attacker can only carry out a MITM attack without the capability of searching for a collision of the digest value. The probability of a successful attack is equivalent to the attack of randomly selecting a digest value (computed using random numbers selected by the attacker). And the attack will be detected if the values do not match. We therefore call it the one-shot guess attack.

collocated, the empirical channel is the face-to-face communication. According to Table A.1, this channel does not have any obvious attacks.

When the customer and the merchant are remote to each other, the options for empirical channel are the use of a *https* web-page or a SMS to display the digest value to the customer. Both channels' security depends on the customer's knowledge about the merchant. According to Table A.1, without the knowledge of the merchant's web-page URL or phone number, their scores are graded as low. We assume that the customer can obtain the merchant's web-page URL or phone number during or before the purchase. This is often true because the merchant is responsible for advertising these kinds of information to accept payments from the customer. According to the attack model, there are security risks in using the two channels. Note that it is flexible to switch between the two channels. And it is unlikely that both channels are compromised at the same time. We therefore put half-block hatchings on V.SSL and V.SMS.

We can prevent Attack (ii) by increasing L. For example, we have mentioned that a standard [20] given by NIST requires that a successful guess of a secret value should be less than one in 1,000,000. This puts the minimum length at 6 decimal digits. However, it is necessary to take into consideration of the value of α which varies when the payment scenario changes. For example, in small payments, in order to improve the speed, it is reasonable to consider using shorter lengths of the digest value, for instance, 4 decimal digits. This decision can only be made by the payment system designer who knows the concrete security and usability requirements.

• Man-in-the-street attacks

The reverse authentication method we use allows customers to keep secrets in their own hands. For example, they can download the authenticated order information from the merchant; they can therefore verify the order information on their own devices and submit their payment information directly to their money provider or to the merchant in the form of an e-cheque; the merchant cannot hold any customer information that can be reused in other payments. This can efficiently reduce the chances of MITS attacks.

6.8.2 Balancing usability and security

[90] examines ways of performing digest 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. Usability tests of the entire solution have not yet been made. However, we can use cash payment as a good reference to help evaluate our solution. For example, cash payments can be made easily at any place without sophisticated knowledge; they are robust in hurried payment scenarios; and they can be made entirely based on context. According to the discussion in Chapter 3, cash payment can achieve the following five security

properties with very low cost of human efforts:

1. The customer clearly knows whom to pay;
2. The customer clearly knows the money has been handed to the intended merchant;
3. The customer clearly knows the amount of the payment;
4. The customer loses, at most, the money being handed over;
5. Anonymity: cash payment does not require either party to know the other's name.

Consider the example of Alice paying Bob. Alice cannot simply keep pressing the OK button regardless of what is displayed on the mobile phone. We force Alice to manually enter the digest value received from Bob. This guarantees that Alice knows whom she is paying.

However, we cannot enforce Alice to correctly check the order information received from Bob. Alice can skip this step and press the confirm button to pay. This leads to vulnerabilities. For example, Bob may display a different name and account number. This may happen when Bob is a malicious shop staff who deliberately replace the shop's name and account with his.

By using a HISP, we can force Alice to look at her phone since she needs to manually copy a digest from Bob and input it into her phone. This regulates Alice's behaviour. Alice can no longer ignore this step. And by careful design, we can enforce, or at least persuade, Alice to check the payment amount and some easy-to-read information about Bob (e.g. a logo or a photo). We conclude this as Advantage 1:

Advantage 1 When human behaviors are difficult to predict and regulate in payments, HISPs can help the standardisation and the regulation of the payment process.

In addition, we can guarantee that Alice loses at most the amount of money described in the order because Bob cannot reuse the payment information received from Alice.

By pressing the confirm button, Alice knows that the money has been handed over to Bob. To ensure that Alice knows how much she is paying, we can force Alice to enter the amount of payment on her mobile phone. To reduce the penalty of manually entering the payment amount on the mobile phone, we can program the application and configure a threshold value β. When the payment amount is less than β, Alice does not need to enter the amount; when the payment amount is greater than β, Alice has to manually enter the amount on her mobile phone. This is only possible when Alice can receive order information electronically from Bob. In comparison to cash payment, our method is better in situations where Alice has to ask for changes from Bob.

Anonymity is optional because in many payments, Bob needs to know where the money comes from. However, when there is a need for anonymity, we can easily implement this. For example, we can choose not to disclose Alice's information to Bob.

We can see that our solution can achieve all five security properties of cash payment by design.

An argument that we learned during our investigation is that the penalty of manually entering the digest value on the mobile phone is sometimes considered to be high. To reduce this penalty, we can design that before the protocol runs, we display two buttons $B1$ and $B2$ on Alice's mobile phone. $B1$ represents payment with amount less than β; $B2$ represents payment with amount greater than β. If $B1$ is pressed, the digest value to be compared is 2–6 digits long; if $B2$ is pressed, the digest value to be compared is more than 6 digits long. The exact length of the digest value to be compared and the value of β are determined by the implementer. This is a balance between usability and security.

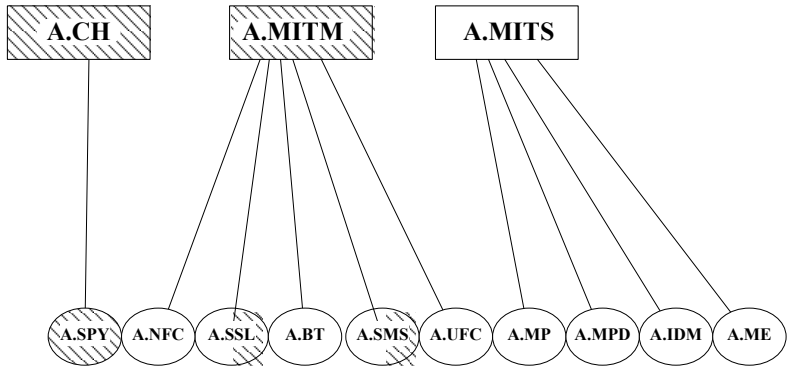


Figure 6.7: The attack graph of HCBK mobile payment solution.

Attack	Likelihood	Impact	Risk
A.SPY	0.5	50	25
		Overall risk:	25

Table 6.1: The security evaluation results of HCBK Mobile Payment Platform when using face-to-face communication.

Attack	Likelihood	Impact	Risk
A.SPY	0.5	50	25
A.SSL	0.5	50	25
		Overall risk:	50

Table 6.2: The security evaluation results of HCBK Mobile Payment Platform when using *https*.

Attack	Likelihood	Impact	Risk
A.SPY	0.5	50	25
A.SMS	0.5	50	25
		Overall risk:	50

Table 6.3: The security evaluation results of HCBK Mobile Payment Platform when using SMS.

6.8.3 Comparative analysis

Figure 6.8 shows the values of the overall risks of all eight mobile payment solutions. In this figure we assume our solution uses face-to-face communication. This is to make it comparable to others; for example, Paypal or Alipay may subject to A.SSL or A.SMS if the customer pays an online merchant (which displays information via SSL connection) or a merchant account received via an SMS.

HCBK Mobile Payment Platform provides the opportunity to develop secure mobile payment solutions with low cost. Firstly, its security does not depend on any existing infrastructure, and the cost of attacking is high because the attacker has to be physically involved in each attack. Secondly, the use of HISPs reduces the effort of security design since we have shifted the burden of defining what is proper context for each payment to the customer: the customer evaluates and decides whether or not to trust the digest received from the empirical channel. Thirdly, we can easily upgrade the security by adding more security information required into the payment process since data are all transmitted electronically; for example, we can add pictures, voices, biometrics and longer random numbers.

It also provides a way of developing flexible applications that can adapt to ad hoc mobile payment scenarios. Firstly, it can be used in payments of any range. This is more flexible and usable than NFC-based solutions. Secondly, it supports all sizes of payments. For example, we can use stronger empirical channels and longer keys to secure macropayments. Thirdly, it makes use of all possible context to facilitate the payment process which gives much freedom for future developers to develop novel payment applications.

In addition, it enables us to conveniently use the mobile banking service to make payment. This can help reduce the number of mobile payment applications to be installed.

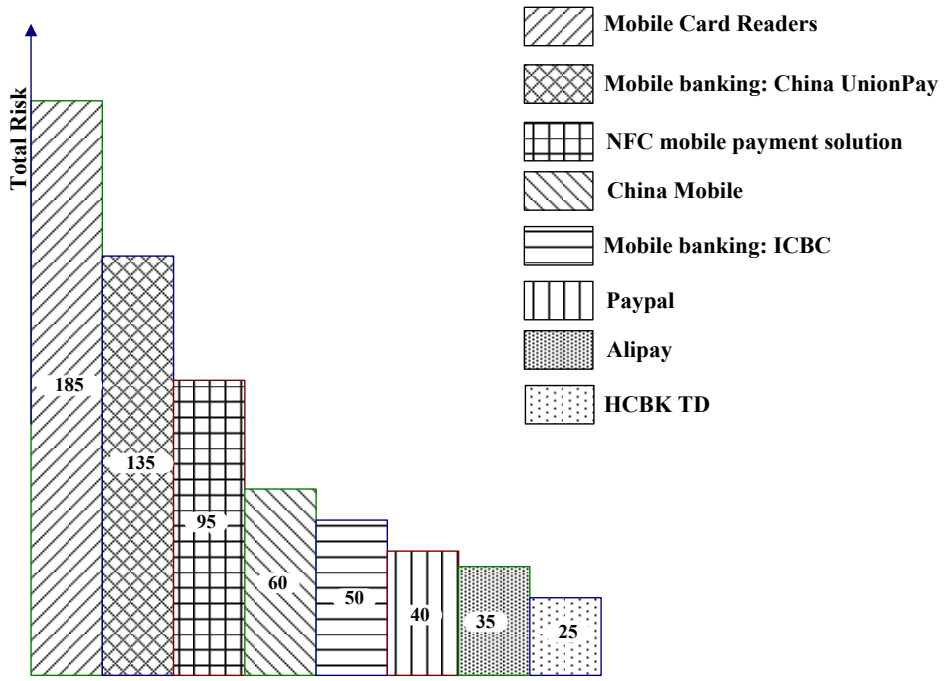


Figure 6.8: Overall risks of eight mobile payment solutions.

6.9 Related research

There are a small number of papers which discuss concrete designs of mobile payment systems, and SA2pMP [160], mFerio [36], MFAMP[8] [29], MP-Auth [103], GOVPKI[9] [77], MobiCash [35], Mobile-to-Mobile Payment System (MMPS) [140], and P2P-Paid [67] are discussed here (see Table 6.4: A comparative view on mobile payment designs).

We notice that the security of SA2pMP, MP-Auth, PKI, MobiCash, and MMPS is based on PKI. PKI is problematic for mobile payment in the sense that knowing that you are paying a large retailer does not tell you that you are paying your own bill. This is not a problem with HISP-based connection, since that will always be run with the specific till, session or other entities that the customer is expecting to pay.

The authors of [103] present a design called MP-Auth which uses mobile phones to protect online banking. Without any use of additional hardware supports, we consider it 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,

[8] [29] describes a multi factor authentication mobile payment solution, and we term it as MFAMP.
[9] [77] describes a solution based on a governmental PKI infrastructure, and we term it as GOVPKI.

Name	Language	Connection	Main Security Mechanism
SA2pMP	J2ME	*http/https*	ECDSA & PKI
mFerio	Not specified	NFC	Fingerprint & NFC
MFAMP	J2ME	SMS	OTP
MP-Auth	J2ME	wire line/Bluetooth	PKI & Password
GOVPKI	J2ME	Bluetooth	PKI
MobiCash	Not specified	Not Specified	PKI
MMPS	Not specified	SMS	PKI
P2P-Paid	J2ME	Bluetooth	Payment authority

Table 6.4: A comparative view on mobile payment designs.

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, for which they use an empirical channel, which is to display a hash result[10] 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 onto 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.

Our solution can simplify the above processes. Since we assume the initial connection as insecure, we can consider the connection between the mobile phone and the PC, and the connection between the PC and the bank, as a single insecure connection between the mobile phone and the bank. This help reduce the complexity when designing security systems.

6.10 Discussion: some interesting questions

The authors in [57] lists a few interesting research questions related to mobile payments. Among which we select and discuss Questions 5–7 under the category of technology. Note that because [57] was written in 2006, its technology background is outdated. For example, the Wireless Application Protocol (WAP) is no longer a popular tool for accessing information online since most modern mobile phones support HTML and use *http/https* to access online services. Therefore, our discussion in this section is mainly based on the mobile payment solutions discussed in this thesis.

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

> Question 5: What are the technological and technology-related strengths and limitations of the main technology 'architectures'?

The current main architectures used to support mobile payments are: (A) e-wallet payment systems (based on mobile applications); (B) mobile banking systems (based on *https* or SMS); (C) and mobile billing systems. We have discussed these in detail and answers to Question 5 can be found in sections of case study in the previous chapter.

> Question 6: What security and trust mechanisms fit various types of mobile payment services?

There are no simple answers to this question. Our investigation does not cover all existing mobile payment solutions. By evaluating and comparing different solutions discussed in this thesis, we conclude that one possible direction is to keep secrets in customers' own hands as opposed to sharing everything with the merchant. This is described as Advantage 0 in Section 1.5.2.

> Question 7: Can standardised transaction protocols and interoperability mechanisms help to solve the roaming problem between networks to facilitate mobile commerce and payment transactions?

We learned during an meeting with China Mobile that the roaming problem does not only relate to technology but also policies and business models. Our solution has somehow bypassed this problem because we do not rely on any existing security of any connection. Current mobile payment solutions mostly do not rely on mobile networks; for example, most of the mobile payment solutions discussed in this thesis can work without using mobile networks except for mobile wallets provided by mobile network operators and SMS-based mobile banking. [83] shows an example solution to Question 7.

6.11 Discussion: pervasive mobile payments in the future

The development of NFC mobile payments gives good examples of what the future of mobile payments would look like. For example, mobile payments can be made to a parking meter, a poster, and an automatic vending machine. As the development of mobile payment continues, the number of pervasive mobile payment types will increase.

However, pervasive mobile payments like these have a serious threat: an attacker can modify or even replace these public facilities; and the number of payment frauds are likely to increase when the number of public facilities with mobile payment capabilities increases. For example, Bob is making an NFC mobile payment to a compromised parking meter; the payment he made is manipulated by the attacker to buy a

much more expensive item in another place. This is a typical MITS attack and it is difficult to detect.

Even if we explicitly request the customer to check the receipt of every payment he/she has made, human complacency is a difficult problem and may make this effort futile.

The challenge is how to improve security without compromising usability in frequent and ad hoc pervasive mobile payments. It is not easy to solve this challenge because different hardware is involved in different situations. And we cannot make the assumption that all these public facilities are tamper-proof, for example, banks are still making efforts to make ATM machines safe after suffering a variety of attacks.

However, we can reduce the complexity of this challenge by using a HISP. When using a HISP, we only need to ensure that the empirical channel is secure. For example, it can be a small display that can be seen by the customer. In this way, we only need to make the display tamper-proof and inform the customer to read the digest from the secure display. The can significantly narrow down the range of protection.

6.12 Conclusion

New technologies have been invented for the purpose of making secure and convenient mobile payments; increasingly more companies and organisations are working together to introduce and promote different mobile payment services in the market. The environment of mobile payments is becoming more dynamic and the situations where mobile payments are made are becoming more ad-hoc.

Keeping secrets in our own hands helps to reduce the risk in such dynamic and ad-hoc environments. Our HCBK Mobile Payment Platform helps to achieve this. When using our own mobile phones, we create a secure connection to the one we need to pay by using a HISP, and since information is transmitted electronically, convenient payments can be made. In this way, we have achieved many strong properties that are otherwise impossible: we do not depend on any existing security, we can use all kinds of electronic connections, we can use all sorts of context to facilitate the payment process, and we can make on-site and remote mobile payments of any amount.

We have discussed and evaluated using HISPs in the payment domain. We have discovered that there are other domains where HISPs can be used. Extending our research in payments to these new domains can help us fully evaluate HISPs. We give a general discussion of these in the next chapter.

Chapter 7

Conclusions and further research

7.1 Introduction

HISPs achieve what one might at first think to be impossible: they bootstrap security over insecure networks such as the Internet and WiFi without any pre-existing network of secrets. They can even authenticate parties to each other who do not know each others' names. They do this via the transfer of a small amount of non-secret information, usually by human users, that is authenticated by context. Based on this fact, we have developed the reverse authentication method dedicated for payments in Chapter 3. This method underpins our solutions of securing payments.

In Chapter 4 our case studies revealed that it is difficult to defend against sophisticated attacks on PCs and the Internet. A dedicated payment device can help to isolate important payment information from PCs. A HISP is used to create the secure connection between the device and the merchant. This secure connection enables the use of an e-cheque which is resistant to MITS attacks. However, using dedicated payment devices requires additional hardware investment. This drives us to the study of payments on pervasive devices like mobile phones.

In Chapter 5 we evaluated different mobile payment solutions and discovered that the pervasive environment where mobile payment takes place creates additional challenges: the traditional threats from online payment attacks on PCs still exist in mobile payments, and new threats exploiting vulnerabilities exposed by new functionalities of different mobile payment applications keep emerging. The challenge is to protect mobile payments in diversified, dynamic and hostile environments.

In Chapter 6 we introduced the concept of a unified mobile payment platform which allows the development of new mobile payment solutions while achieving strong security. This platform accomplishes three functions: making the initial insecure connection, securing the connection and completing the payment. In order to improve usability, we developed a new naming strategy which allows the use of many possible information to speed up the establishment of the initial insecure connection. This strategy is only effective when a HISP is used.

The main motive behind the use of HISPs is to solve different security requirements in pervasive environments where traditional security like PKI is not flexible and economical. Recently we started to study using HISPs in other domains. We will give an introduction of this study in the next section.

7.2 Using HISPs in other domains

We observe a HISP has the following two features: (A) using human interactions; (B) using two channels. The first feature indicates the nature of using a HISP: it relies on the human trust which allows us to carry out authentication without using any existing security infrastructure. The second feature indicates the process of using a HISP: we first set up normal electronic connections among devices, and then use empirical channels among humans. We may further conclude that a HISP requires at least two elements: (i) electronic devices (at least two); (ii) human presence (at least one).

These features immediately introduce the following challenge: the cost of total human effort and the complexity of the network is in direct proportion to the number of participants. This may limit the practicability of using HISPs in other domains where bootstrapping security for a large group is necessary.

In payments, we can construct a general security model since pair-wise payment relationships are easy to define; for example, peer-to-peer payments and customer-to-merchant payments. However, when there are multiple parties, and when there are multiple different relationships, it is difficult to generalise various scenarios into a single model, therefore, it is difficult to find a single solution to solve the challenge stated earlier.

We select two cases where we believe HISPs can be used. The first case is medical sensor network where a human wears a set of on-body-sensors (OBSs) which are constantly monitoring his/her health. The second case is group authentication. Instead of focusing on small collocated groups, we choose to study how to use a HISP to bootstrap a large group when group members are collocated, not collocated, or the mix of the former two situations. This case study can be therefore used as references for more complex situations.

7.2.1 medical sensor network

We focus on finding a solution to allow an OBS to securely deliver medical data that it sensed to a Healthcare Provider (HP). Traditional design of ad-hoc sensor networks are not suitable in this case because OBSs that can be deployed on a human body are in small numbers and close to each other. A much cheaper choice is to use the patient's own mobile device to transfer data via telephony or WiFi. We call this system Mobile Medical Monitoring System (MMMS). A MMMS is a security critical system. The challenge is that there is no existing security between a patient and a

HP, and the relation between them is dynamic: a patient may frequently change his or her HP. Therefore we have to bootstrap security "on the fly".

We can divide the security domain of a MMMS into two: a patient area domain (PAD) and a medical server domain (MSD). OBSs are grouped within a PAD where the patient has absolute control, and patients are grouped within a MSD which is controlled by the medical server or the HP. The security of MMMS should allow a patient to easily change his or her OBSs or HPs. For example, the patient may choose which OBS to wear on different days, and the patient can purchase new OBSs and add them to the existing PAD. A patient can also change the MSD he or she belongs to, if, for example, the patient moves to another city, or if the patient chooses a different HP with better services. Therefore, the associations between an OBS and a PAD, or between a PAD and a MSD, can change from time to time.

To achieve the above security requirements, we propose using a HISP to bootstrap security for MMMS. We assume a set of OBSs which all have a short-range wireless connection. These OBSs are connected to a mobile device, which then transmits medical data to HP. (See an example design of the system structure in Fig 7.1).

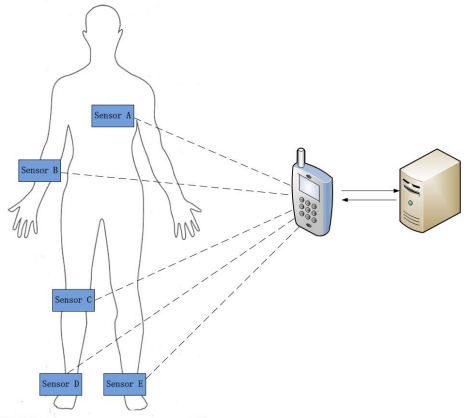


Figure 7.1: OBSs and connections.

When using a pair-wise HISP, as we have demonstrated in Chapter 4, we can shift the computation consuming cryptographic functions to the mobile device. For example, the mobile device will generate the public key pair and send the public key to the OBS; the OBS will only need to compute encryption using the public key and send back the result; the mobile device then perform decryption using the private key.

Since the range of the two connections (one between the OBS and the mobile device; the other between the mobile device and the HP) is different, it is possible to use different security strength for different connections. In other words, since the range of wireless communication on an OBS is much shorter than that of a mobile device, we may assume that the space of opportunities for an attacker to attack is

accordingly smaller in a PAD than that in a MSD. This breaks the security of a MMMS into two levels. We can, therefore, consider sending a shorter public key from the mobile device to the sensor, and using a shorter symmetric key as their session key.

Recall the two features of using a HISP: (A) using human interactions; (B) using two channels. In a PAD, the advantage of using a HISP is clear: we can easily use human interactions to establish the empirical channel since the distance between a sensor and a mobile device is very close. A human can conveniently touch and handle either a sensor or a mobile device.

However, a normal sensor often does not have such interfaces except for three LED lights (red, yellow and green), a simple buzzer and a press button.

Our research[1] of creating an empirical channel between a sensor and a mobile device focuses on the following methods:

1. A straightforward method is to display the digest value on the OLED display, however, in practice, ordinary small sensors often do not have such a display.

2. Using the LEDs to display the value. We have added two more LEDs onto the sensor. Our initial tests show that reading flashings by humans does not produce accurate results. Others propose using a camera to capture and decode the "flashings" of LEDs [142, 141].

3. Enabling the mobile device to "hear" information from the sensor. Examples of related research can be found in [70, 145].

4. Creating new empirical channels between an OBS and a mobile device based upon particular applications.

Since there are no strict standards on developing hardware or software on sensors and mobile devices, all four methods presented above are usable depending on context. Details of our research can be found in [82, 80, 81].

7.2.2 Group authentication

A HISP, as its name suggests, always involves human interactions. We understand these interactions are either accomplished by making physical interactions (when collocated) or via proxies (when remote).

When using a group HISP, to symmetrically compare SASs in a group of size N ($N > 2$), a total number of $N(N-1)/2$ two-way interactions (for example, conversation via phone calls), or $N(N-1)$ one way interactions (for example, sending text messages or taking photos of 2D barcodes), have to be made. Given $N = 8$, there can be 28 – 56 interactions. To reduce the cost of human effort, a possible solution is to: (i) reduce the number of human interactions; (ii) reduce the human effort of each

[1] This is a joint research project with Xin Huang from Oxford Martin School.

interaction. In addition, the cost of human effort and the difficulty of bootstrapping a group will increase when group members are remote to each other.

Only a small amount of practical research has been reported on implementing group authentication HISPs. Some good examples are [52] and [99]. [52] exploits the Seeing-is-Believing [107] technique to bootstrap a large group (size from 2 to 25) by taking pictures of 2D Barcodes displayed on mobile phones; [99] proposes a new technique of displaying comparable images on mobile phones to bootstrap a small group (size from 2 to 8). These solutions are usable when all group members are collocated; for example, when they are in the same room.

We are not aware of any comparable published research for bootstrapping a group remotely by using HISPs, and existing research does not include large (e.g. $N > 100$) groups. This may reveal a critical problem: HISPs, as they require empirical channels which always involve human interactions, are mostly convenient when humans are close to each other, and the total amount of human effort is in direct proportion to the size of the group.

This is probably only feasible for large groups, where it is possible for parties to *broadcast* information to each other empirically: for example, by speaking at a meeting. Speaking aloud in the public is a typical one-to-many empirical channel. We will continue our discussion in the next section.

• Introducing group HISPs

The Symmetric HCBK (SHCBK) protocol [121] is a typical group 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. $\forall A \longrightarrow_E \forall A' :$ users compare $digest(hk^*, INFOS)$, 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[2] 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. A careful security analysis of this protocol (see [121], 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 different views of the authenticated information.

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

Good HISPs such as SHCBK therefore offer maximum security for a given amount of human effort.

We can reduce the number of human interactions if there is a trustworthy Initiator I, consider the rest of the group as G', then the above protocol can be modified as following: in the process of comparing digest values, I compares digest values published by other group members; others compare the digest value published by I; I then publishes the final result of digest comparison; others checks this result. We call it Semi-SHCBK protocol. Therefore the total number of messages to be exchanged via empirical channels changes from $N(N-1)/2$ to $3N-3$. If there is a trustworthy Initiator, when $N > 6$, Semi-SHCBK protocol is more efficient than SHCBK protocol.

Recall the example of the one-to-many empirical channel in the previous section, by using it, we can reduce the cost from $3N-3$ to $N+1$. For example, I compares digest values published by others, I reads out his/her digest value to others; I then reads out the final result of digest comparison. Other examples of one-to-many empirical channel can be writing[3] down the digest value on a blackboard, or speaking[4] on the TV or radio.

Clearly, when there is a trustworthy Initiator and an one-to-many empirical channel, the efficiency of using a group HISP can be significantly improved. However, in practice, one-to-many empirical channels can be difficult to establish. For example, the voice of the Initiator may not be heard by all group members, group members are not in the same room, or group members cannot recognise or trust the Initiator. Therefore, such one-to-many empirical channels cannot be considered as general solutions to improve the usability of HISPs.

Remote authentication is made by using a proxy, or a "referee". It is a common practice in our daily life that when P_1 meets an important person P_2, presenting a reference letter from a person P_3 that P_2 knows can effectively earn P_2's trust. Therefore the trust from P_2 between P_3 can be used to bootstrap trust between P_2 and P_1, even if P_2 has never met P_1. The creation and delivery of such a reference letter is essentially the communication via an empirical channel. For example, Bob makes a phone call to Alice; by recognizing the caller's phone number or his voice, Alice knows that she is communicating with Bob. In this case, the telephone is a proxy between Bob and Alice. However, the assumption of knowing the caller's phone number or recognising his voice does not stand in ad-hoc scenarios.

The difficulty of remote authentication arises when two parties cannot easily "find" each other; for example, Bob may not know Alice's phone number, or Alice does not recognise Bob's phone number.

An additional challenge is coordination. For example, how is it possible to organise human interactions to improve efficiency as well as to reduce security mistakes?

[3]The requirement for this is that group members have to make sure it is the Initiator who has written down the digest value.

[4]The requirement for this is that group members can recognise the Initiator's face or voice.

In order to solve these challenges, we introduce the idea of using social networks to bootstrap security for a large group in the next section. The purpose is to find a communication channel that facilitate group formation before running a HISP and the digest comparison when running a HISP.

7.2.3 Social networks for importing and exporting security

Given that the social network providers are increasingly making their applications available as secure web-sites, there remain two primary concerns:

A How can we know that a given site belongs to a given user: the *identification*, or *authentication* problem? In general such knowledge may be absolute or come with some identifed confidence level.

B The provision of appropriate security models for collecting, using and sharing data from the local user and his or her devices including sensors.

In this section we concentrate on A, and furthermore show how security developed for social networking can be used to conveniently bootstrap other secure connections.

We imagine that in general solutions to A might involve any one, or combinations of (i) prexisting security infrastructures such as PKIs, (ii) reputational models based on trust ratings by other network users, and (iii) bootstrapping security by person-to-person contact by interaction outside the social network. In this paper we concentrate on (iii) and show how HISPs can be used to do this efficiently when there is a means for getting a small amount of information from the owner of the page that is to be authenticated to the person who wants to authenticate it. This transmission might be via personal contact or using a second medium that is trusted as authentic.

HISPs can be thought of as tools that enable one (perhaps informal) authentic channel to efficiently authenticate, and then secure another one. This means that they have two complementary potential uses in social networking.

1. We can use a HISP to authenticate online identities by using existing connections (typically personal or telephone conversations between the humans involved.) In this case, we import security from existing social relationships to social networks.

2. We can use a HISP to create secure connections between devices, in this case, we can use authenticated social network accounts as proxies to display SAS's. This can significantly improve the usability of HISPs. We therefore export security from social networks to other applications.

• Proving online identities

In order to use OSNs as empirical channels we must answer the following question: "*how do I know that what I am seeing on the page comes from the person or other*

entity that I think it does". To better analyse this problem, we divide it into two sub-questions: how do I know the (e.g. Facebook) page I am seeing is authentic within the OSN? and how do I know it belongs to the person I think it does? The first of these questions can be solved by conventional computer security, for example, the *https* service on OSNs.

The second question can be converted into the following one: "is this an established Friend for which you are certain of the link between page and person?" If the answer is yes, then secure access to that page is clearly a good empirical channel. This is the most common way of authentication in our daily life. For example, one may have experiences in interacting with a social network account, one may authenticate a social network account by the number of common Friends, or one can authenticate a social network account by viewing its profile, Friends list, photos, history of participated events and other context information.

If we cannot make our decision based on past experiences, we may use telephony or physical interactions to accomplish this task. A HISP can therefore be used to authenticate OSN accounts. For example, Alice wants to know that the social network account of Bob is authentic; if Alice has a phone number of Bob and she is certain of the authenticity of this phone number, she then runs a HISP with Bob to verify his account by using telephony as the empirical channel.

Note that the availability of HISPs provides us with the flexibility to bootstrap security from any existing authentic connection, whether one derived from physical proximity or other means such as telephony.

And there are other alternatives of authenticating online identities in practice, for example:

1. Centralised authentication. For example, Twitter provides authentication service. The verified account will display a special indicator (a small icon or a "badge"). However this service is limited to celebrities on Twitter. A similar situation can be found in other OSNs.

2. Introducing decentralised authorities. For example, we can publish OSN accounts of a group on a company's *https* web-page. In this case, the company acts as an authority which authenticates a group. Similarly, a trusted organisation or a trusted individual can also play the role of an authority. For example, a community leader may only keep Friends that belong to the community, therefore his or her Friend-list can be used to help authenticate the community members. This can be used to replace the human effort of authenticating group members and can greatly improve the application in authenticating a group when its size is large. In our implementation, when prompting users to verify the member-list of a group, we provide an option for users to use a trusted authority (in the form of an *https* web-page).

3. Introducing trust ratings. Rating by trust is a common practice in OSN research, for example, the authors of [69] describe a semantic web-based OSN,

and they developed algorithms to rate the inferred reputation of a node. Another distinct example is PGP. It exploits ratings to determine the level of authenticity of downloaded public keys. A rating scale of 1 to 4 is used: full (complete trust), marginal (partial trust), untrustworthy and don't know. The most distinct advantage of this method is that it provides pervasive automated authentication. We have implemented a demonstration rating system by using the same ratings introduced in PGP.

4. Blackballing. Blackballing[5] is a voting method used in many gentleman's clubs: members have a large number of white and black balls and each member casts a single ball into the ballot box to vote for a proposition, if there are one or more black balls in the ballot box, everyone will immediately know this proposition has been vetoed. In our implementation, each member checks the list objects one-by-one, if one object is "vetoed" by one member, then list L is "vetoed". This is also a form of utilising "crowd knowledge" which effectively reduces the security mistakes when members manually authenticate each other. Note that this is likely to be an effective way of ensuring dubious parties are identified, at the expenses of loss-of-service attacks.

• Bootstrapping a large group by using OSNs

The correctness of bootstrapping a group can be defined as follows: all members acknowledge a list L, which contains details of all members; the resulting group G contains exactly the same number of members recorded in L and no one, except for the members included in L, can be allowed to join G. To fulfill this task, we need to identify and overcome the following challenges:

- Collecting group information. This is to create list L.
- Counting and authentication. Counting is to check whether the size of group G matches with the size of list L. Authentication is to check whether members included in list L are legitimate. This is used to detect two types of attacks: (i) MITM or outsider attacks; (ii) Sybil [60] or insider attacks.

Counting, if made by humans, has limitations. For example, one may make mistakes when the size of the group is large. GAnGs [52] assumes humans can accurately count less than ten individuals via physical interactions. They randomly divide a large group into small subgroups in order to allow humans to count and verify members correctly. This action provides greater usability but leads to weaker security: there may be the chance that attackers are allocated to the same subgroup. The probability of attack detection [52] is less than the value of $1 - 2^{-b}$ assumed by the HISP (b is the bit-length of the SAS). In addition, subgrouping can be laborious and inconvenient since it has to be randomised.

[5] `http://en.wikipedia.org/wiki/Blackballing`

Authentication normally requires more human efforts. For example, in [52] they use visual channels (created by mobile phone display screens and cameras) to check the presence of identities. Since visual channels of reading 2D barcodes on mobile phones are normally unidirectional, a symmetric authentication of a group of size N requires $N(N-1)/2$ interactions. This number increases quickly when the size of the group increases.

When using OSNs as empirical channels, we can first divide the entire process into the following two steps:

1. Authenticate OSN accounts included in list L are legitimate. We call it the authentication of online identities.

2. Run the HISP. Read and compare digest values displayed on members' OSN pages. We call it the authentication of connections.

Step 1 is to ensure that we can use OSN accounts as proxies of our physical presences. Step 2 is to test the presence of MITM attackers by using a HISP, which is to authenticate that the electronic connection is correctly connected to the intended device represented by the OSN account. This strategy can remove the requirement for physical interactions in Step 2.

In Section 7.2.3 we have discussed various techniques of proving online identities. These allow humans to conveniently and efficiently adapt their authentication strategy according to different scenarios. In addition, we can conveniently run a program to automatically count and check whether the number of responsive[6] (or active) OSN accounts is equal to the number of accounts included in list L. We therefore conclude that counting is unnecessary in our solution and there is no need of subgrouping. This improves both security and usability.

More importantly, once we have authenticated that OSN accounts included in list L are legitimate, Step 2 can be made automatically since we use OSN accounts as proxies of our physical presences. This is a significant improvement which provides more capacity for large groups, for example, groups with size over 100. In addition, we can display long digest values without increasing the cost of human efforts.

It is worth noticing that on OSNs, the cost of Sybil attacks of creating multiple fake accounts are higher because OSN providers, for example, Facebook[7], Google+, require unique identifiers (email addresses or mobile phone numbers) to register, and they keep records of online interactions which can be used as indicators of authenticity.

We also notice that by reducing physical interactions, we can reduce impacts from other uncontrolled factors; for example, the luminous intensity, the physical distances, the quality of mobile phone cameras or display screens, and most importantly, the human complacency.

[6]Those who display the digest value.

[7]In our investigation, we discover that Facebook normally requires at least one mobile phone number to register; and accounts registered by email addresses will later be required to be authenticated via a mobile phone number.

Another significant improvement in usability may be that we can allow delayed running of HISPs. Experiments of relying on physical interactions to run HISPs have one implicit assumption that all members have to finish the process of authentication within a short period of time. And it is the reason that reducing time is critical for improving usability. However, in practice, the cost of coordination can be high and humans may not necessarily be available of carrying out the same physical action at the same time. This problem can be more significant[8] when humans are remote to each other. By using OSN accounts as proxies of our physical presences, we can divide the authentication process into two separate steps discussed earlier in this section. Because Step 2 of reading and comparing digest values can be automatically completed by using a program, and Step 1 of authenticating the legitimacy of OSN accounts in list L can be carried out asynchronously, the running of HISPs can be delayed until the last member completes the authentication in Step 1.

This allows more useful security applications. For example, a department sends out notifications of bootstrapping a secure network for internal communication to its employees. Some employees are travelling abroad and they are not responding immediately. By using our solution, the program keeps waiting until the last employee responds which triggers the authentication process. After the authentication process has been finished, the program displays results to all employees.

The model of social networks for importing and exporting can be used to support a wide range of security applications. For example, secure location sharing, secure online communications or secure Cloud Computing. See Appendix B for a demonstration implementation of secure location sharing on mobile devices. [51] shows more details of this research.

7.3 Unresolved research questions

This section discusses two unresolved research questions. The goal is to improve our study of using HISPs in practice; for example, the first proposal is to establish a more complete risk evaluation model for payment systems; the second proposal is to understand how to manage security for a dynamic large group.

7.3.1 A complete risk evaluation model for payment systems

A complete payment system often includes four components: the customer payment system, the merchant payment system, the e-wallet payment system, and the banking system. We have simplified our risk evaluation model and our analysis mainly focuses on the critical part of the first two components. This makes our analysis appear to be insufficient if we need to understand all potential risks within a specific payment system.

[8] Our experiment shows evidence of high cost of coordination.

Since risk evaluation requires understanding of the payment system and opinions from both the technical side (e.g. IT support, programmers and system designers) and the non-technical side (e.g. customers, merchants and managers from banks), establishing a complete risk evaluation model for payments is a sophisticated and long-term project. More in-depth investigations and surveys need to be made in order to collect data from the payment industry. In addition, study on payment behaviours is necessary since we have already revealed that human complacency can disable well designed security in payments.

7.3.2 Group formation

One of the most important research topics, related both to security models and underlying technology, is group formation. Technologically, the challenge is to establish and maintain a sufficiently large ad hoc group of devices using the connection technology available in a given scenario.

Should a group be formed using a single initiator, a tree structure, broadcasting over a fully connected graph, or some other topology? This question can be posed both for initial network formation and for digest comparison between humans. For example, when creating a large group, often we may not be able to communicate with all members at the same time. For example, some members may be in remote areas that are not currently reachable; or new members may come to join the group when only part of the group is available. Then one or more members may be delegated (e.g. by a group leader or via vouching by other members) to complete group formation. In general, this is to delegate trust to someone we believe is trustworthy. We need to develop a robust trust management framework that can guarantee the standard of security is not undermined during delegation of group formation.

How should we manage group amalgamation and splitting, and adding or removing a single member? For example, when adding a new member, he/she may only be allowed to temporarily communicate with others. After a certain period of time (e.g. after he/she finished his/her task), he/she will be automatically disconnected and removed from the group. This introduces a difficult challenge to key management. For example, permanent addition may involve key sharing, for instance, one may share the existing group key with the new member. Or the existing group may run a new protocol with the new member to generate new keys. Addition that merges different groups may require a hierarchical key structure since communication may be open among groups but it may still need to remain secret within a group. Temporal addition may need to generate a new key that is only valid for a short period. Control of these temporal keys may require a decentralised authority that can issue new keys with time validation. Reduction requires policies and techniques for key revocation. This can be difficult since in ad hoc scenarios, not all members are uniformly within the communication network at the same time. Failure or delay of key revocation may lead to immediate security breach.

7.4 On-going projects

We introduce a few on-going projects of using HISPs in practice in this section.

7.4.1 Secure payment

We are continuing our research on payment in order to: (i) investigate more cases of payment systems; (ii) collect more information about the middle end merchant payment system and the back end banking system; (iii) conduct tests to study payment behaviours in different scenarios; (iv) establish a complete risk evaluation model for payment systems; and (v) develop new payment applications. This project is helped by ISIS-Innovation[9] in Oxford.

7.4.2 Secure medical sensor network

Our aim is to develop a secure system that can be used for medical sensors in remote medical care applications. The current focus is to develop convenient but secure methods of exchanging the digest value between a sensor and a mobile device. We are developing methods of (i) using mobile devices' cameras to read information displayed by LEDs on sensors and (ii) developing new empirical channels based upon particular applications.

7.4.3 Secure communication in disasters

Disasters, like the Japanese Tsunami of 2011, can destroy communication infrastructures as well as security infrastructures. In such events, establishing communications is a vital and time-critical job – one which cannot ignore the importance of security. Rescue operations often require cooperation among forces from different sectors, for example, police, fire services, ambulances, military, civilians and local authorities. Security is crucial to guarantee the integrity, confidentiality and availability of communication channels in these operations. Given the difficult and unpredictable context of use, it is useful to have a simple and straightforward means of establishing an ad-hoc security infrastructure that connects different communication channels from different security domains without depending on usual communications or security infrastructures.

Our project is to provide a solution that can quickly and securely establish a usable wireless communication system in disaster scenarios. We have implemented a demonstration system which allows mobile phone users to create an ad-hoc network via the WiFi[10] capabilities of their devices. It currently supports location broadcasting which helps the search and rescue operation. The next stage is to implement our dynamic naming strategy on top of the routing protocol. This is to create a semantic

[9]ISIS-Innovation is a technology transfer company wholly owned by Oxford University.

[10]The WiFi we use is in ad hoc mode.

layer which allows searching using human-meaningful information in mobile ad-hoc networks. HISPs are used to detect any attacker presence.

7.4.4 Secure inter-organisation communication

The Leveson Inquiry[11] has revealed a critical security challenge: public communication media cannot be trusted and attackers can intercept and read sensitive information even though it is supposed to be confidential. The widespread misuse that has come to light in the Inquiry provides a clear reminder of the importance of security in both the public and private sector. Facilitating efficient and secure communications within these organisations, however, is an ongoing challenge.

Existing security infrastructures are frequently too rigid (e.g. might be constrained to only one part of an organization) and cumbersome to support day-to-day interactions, or inter-organisation communications.

A more flexible approach to security would allow the protection of daily communications such as emails, text messages, phone calls, and file exchanges in a tailored and ad-hoc manner. This would enable, for example, people from one office to share data and communicate securely with another over a secured WiFi or 3G network, as might best fit their working needs.

Our project is to provide a generic secure communication system which can be used in the above scenarios. A prototype system has been made. It currently supports sending emails and SMSs, making phone calls and sending files.

7.5 Conclusions

We have discussed solutions of using HISPs to secure payments. HISPs first provide a convenient way of connecting devices, which allows the customer to use their own devices to make payment, and information can be transferred electronically. This improves both security and usability since the merchant will be authenticated before the payment take place and the customer can download the order information from the merchant to his/her own device. HISPs also provide opportunities for innovations in payment methods; for example, we can apply a dynamic naming strategy which allows binding a wide range of identities to the payment account; we can utilise the growing sensibility of mobile devices to incorporate more context into payments; and we can reduce human complacency by forcing the customer to manually enter the digest received from the merchant.

Our investigations into the payment industry are essential to our research since payment is traditionally a restricted area where only limited information is exposed to academics. Our case studies show there are various risks in the existing payment solutions; some are extremely dangerous and urgent attention is required. Our payment solutions provide cost-effective alternatives to these problems since we can shift

[11] `http://www.levesoninquiry.org.uk/`

the burden of authenticating the merchant from the system to the customer. HISPs provide new opportunities for banks as well. We have discussed in Chapter 6 that a unified payment solution is preferred in the future. A bank can easily convert its mobile banking application into a flexible mobile payment application by using a HISP.

In the future we envisage mobile payment will dominate the micropayment market. The customer will be encouraged to use their own mobile devices to make payments. However, it is highly possible to see a rise of attacks against mobile payments since they are dynamic and pervasive. For example, according to our case study in Chapter 5, pervasive payment technologies like NFC mobile payment and mobile card readers does not only introduce convenient payment methods to the customer, but also new attacks. While there are efforts of making mobile payment secure, human complacency frequently makes these efforts ineffective. Research on human factors in mobile payment scenarios is needed in the future to better regulate the payment process and hence improving security.

Macropayment, on the other hand, often requires dedicated systems and devices to guarantee strong security. For example, in Chapter 5 we have indicated that having effective measures against mobile malware can be difficult because of the dynamic mobile phone market and the increasingly sophisticated mobile OSs; in Chapter 4 we have pointed out that payments on PCs suffer from many attacks which are difficult to detect and prevent. This, again, proves our theory of keeping secrets in the customer's own hands. Our implementation in Chapter 4 shows that a cheap card reader with USB connectivity is possible to allow secure payments by using a HISP.

One additional challenge, which makes our research appear to be insufficient, is that the payment market is constantly changing. For example, recently we read news that banks are stepping into the mobile wallet market [48]. This has been recommended in our study since it can consolidate the payment process. However, the competition between banks and mobile payment companies makes the development of the mobile payment market more unpredictable. More case studies are needed in the future to identify risks of emerging new mobile payment solutions. For example, one interesting question would be: how to revoke a mobile wallet if we lost a mobile phone?

HISPs are not limited to payment. Our research shows that they are efficient in other domains; for example, bootstrapping security for a group by using OSNs. The concept of "human interaction" has already been extended from offline physical interactions to online virtual interactions. The idea of "quantifiable-self" is becoming true thanks to the increasing sensibility on mobile devices. The challenge is that how to incorporate pervasive online trust in HISPs. Our research provides a simple solution of importing and exporting security on OSNs by using HISPs. However, this is only a start of an ongoing project. More in-depth research is needed to compute the quality of online proxies, or our virtual presences; many research questions are to be answered, for example, when and how can online trust become transitive? or what essential factors (e.g. frequency of interactions) can be used to authenticate an

online identity?

In the future, we will continue our research on practices of HISPs. For example, besides the payment project, we have ongoing projects on OSNs, medical sensor networks, communication in disasters and inter-organisation communication. By examining different scenarios, we can produce more security patterns of using HISPs, which we believe will become useful references for research in this area.

Appendix A

The guideline of using empirical channels

Table A.1: The guideline of using empirical channels.

Channel	Stimulus	Knowledge	Score	Notes
Face-to-face[1](A)	Audio&Visual	The payer knows[2] the payee before the payment.	High+	This is perhaps the strongest empirical channel. It is difficult to impersonate a person in real life. The better the payer knows the payee, the more difficult for an attacker to impersonate the payee.
Face-to-face(B)	Audio&Visual	The payer does not know the payee before the payment.	High	An attacker can commit a fraud by impersonating the payee. But the payer is highly possible to pay only after he/she has received the goods in face-to-face scenarios. The loss is likely to be limited to the payee's side.
Phone call(A)	Audio&Visual[3]	The payer knows the payee's phone number and voice.	High-	It is difficult to impersonate one's voice or to hack the telephone network. But some mobile phone viruses may be able to hijack the mobile phone connection.
Phone call(B)	Audio&Visual	The payer only knows the payee's voice.	Medium+	It is difficult to impersonate one's voice.
Phone call(C)	Audio&Visual	The payer only knows the payee's phone number.	Medium	It is difficult to hack the telephone network. But some mobile phone viruses may be able to hijack the mobile phone connection.
Continued on next page				

[1]Face-to-face communication here means the payer can both see and hear the payee.
[2]We say one knows one another means that one knows one another's voice, image and perhaps other physical information.
[3]The visual stimulus comes from the reading of the payee's phone number.

Table A.1 – continued from previous page

Channel	Stimulus	Knowledge	Score	Notes
Phone call(D)	Audio&Visual	The payer knows neither the payee's phone number nor his/her voice.	Low+	An attacker can make a phone call to the payer and claim to be the payee. But the cost of this attack is relatively higher than online attacks since it requires the attacker to make the phone call.
SMS(A)	Visual	The payer knows the payee's phone number.	Medium	It is difficult to hack the telephone network. But some mobile phone viruses may be able to hijack the mobile phone connection.
SMS(B)	Visual	The payer does not know the payee's phone number.	Low	An attacker can easily send an SMS to the payer and claim to be the payee. But the cost is still relatively higher than online attacks since it requires the access to the telephone networks.
Voice call[4](A)	Audio&Visual	The payer knows the payee's call ID and voice.	Medium+	It is difficult to impersonate one's voice. But the security of a VoIP network depends on whether or not any encryption is used to secure the data connection. We assume that the security of VoIP networks is weaker than telephone networks. And there are some viruses capable of hijacking the voice call data connection.
Voice call(B)	Audio&Visual	The payer only knows the payee's voice.	Medium	It is difficult to impersonate one's voice.

Continued on next page

[4]Voice call refers to VoIP voice call.

Table A.1 – continued from previous page

Channel	Stimulus	Knowledge	Score	Notes
Voice call(C)	Audio&Visual	The payer only knows the payee's call ID.	Medium-	It is difficult to hack a VoIP network. We have indicated that this depends on the security of the VoIP network used, and we need to consider threats from viruses.
Voice call(D)	Audio&Visual	The payer knows neither the payee's call ID nor his/her voice.	Low+	An attacker can easily make a voice to the payer and claim to be the payee. This however will require the attacker to make the voice call by itself.
Video call[5](A)	Audio&Visual	The payer knows the payee's call ID and the payee himself/herself.	High	It is difficult to impersonate a person. But there is no guarantee that the data connection is secure unless it is properly encrypted. Also there are some viruses capable of hijacking the video call data connection.
Video call(B)	Audio&Visual	The payer knows the payee.	High	It is difficult to impersonate a person.
Video call(C)	Audio&Visual	The payer only knows the payee's call ID.	Medium-	There is no guarantee that the data connection is secure unless it is properly encrypted. And we need to consider threats from viruses.
Video call(D)	Audio&Visual	The payer knows neither the payee's call ID nor himself/herself.	Low+	An attacker can easily make a video call to the payer and claim to be the payee. This however will require the attacker to make the video call by itself.

Continued on next page

[5]Video call here is made via the Internet.

Table A.1 – continued from previous page

Channel	Stimulus	Knowledge	Score	Notes
SN[6] message(A)	Visual	The payer knows[7] the payee's social network account.	Medium	Social networks usually provide rich context information about one person, and one can tell whether the message comes from the right person or not by viewing the context information on his/her page. However, the security of this channel also depends on the security of the social network used, and we also need to consider threats from viruses.
SN message(B)	Visual	The payer does not know the payee's social network account.	Low	An attacker can easily set up a social network account and claim to be the payee.
Https page(A)	Visual	The payer knows[8] the public key certificate of the *https* web-page.	Medium	The security of a *https* web-page depends on the Public Key Infrastructure. Its security is reasonably strong given banks are currently using it in online banking services. But there are some viruses that can hijack the *https* connection or compromise the web-browser.

Continued on next page

[6]SN stands for Social Network.

[7]We say one knows one another's social network account means that one knows that a certain piece of information displayed on a certain social network page comes from the right person. In other words, one knows that a social network account is not fake.

[8]We say one knows a *https* web-page's public key certificate means that one can make sure that the web-page's address is associated with the public key certificate downloaded. In other words, one knows that the information displayed on a *https* web-page comes from the right entity.

Table A.1 – continued from previous page

Channel	Stimulus	Knowledge	Score	Notes
Https page(B)	Visual	The payer does not know the public key certificate of the *https* web-page.	Low	An attacker can easily set up a *https* web-page and claim to be the payee.

Appendix B

Implementation of secure location sharing on social networks

We have implemented a secure location sharing service to demonstrate the use of our security model. We have developed three versions of mobile applications: RIM (Blackberry), Android, and iOS (iPhone, iPad and iTouch). One server SO is used as the coordination server. All devices are connected to SO. To demonstrate the use of on-device computing resources, we have also implemented a demonstration storage directory service (we call it CS Department Internal Cloud).

The mobile phone application first checks the ratings; if there are accounts which fail to pass the rating check, it will prompt the user with a dialogue calling for authentication resource (from a decentralised authority), it will automatically remove the authenticated objects from the stack of the member list; the objects that are left on the stack will be verified by empirical authentication, for example, by using a HISP. Figure B.1 shows the flow chart of the authentication process.

If the entire member list has been verified, the protocol starts to run; the user will then start to share his or her data of locations on Facebook (or directly between devices) if the protocol has been finished successfully; the user can also manage his or her shared on-device storage. Figure B.2 shows the screen shots of the application on Android.

An OSN account is created by the Initiator for the group. This group account is shared within the group, therefore each member can access this account and publish their encrypted data anonymously using the anonymous ID. This message is constructed in the form of (Anonymous ID, Encrypted Message, MAC_G). The last component is a MAC using the group key.

We use Bouncy Castle Crypto Java API on RIM and Android; and OpenSSL C Library for iOS. We use 1024-bit Diffie-Hellman public keys to generate shared secret keys; 128-bit AES is used to encrypt data.

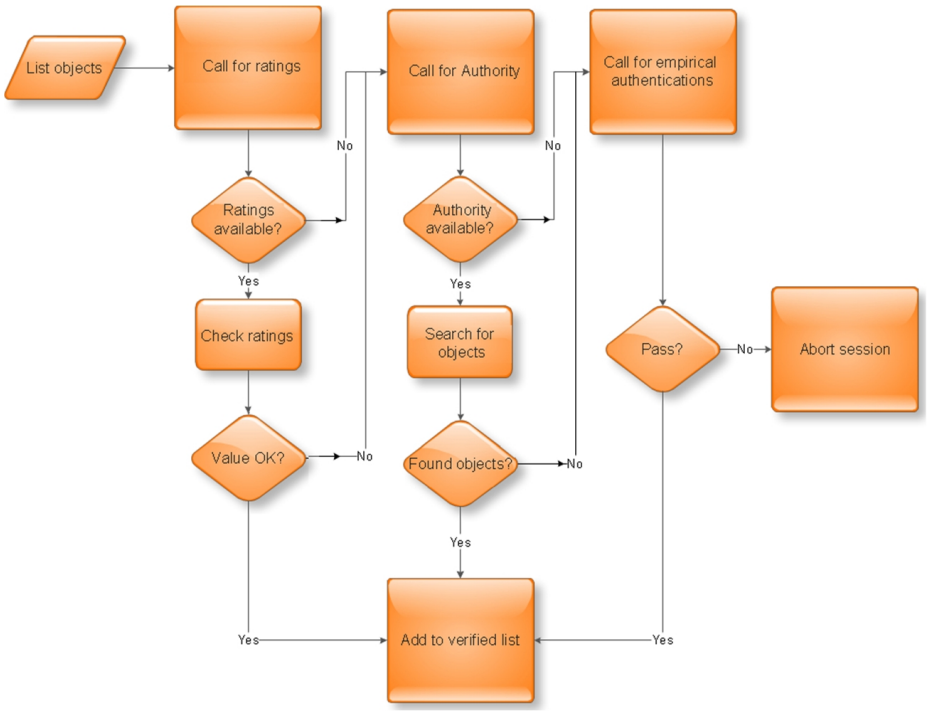


Figure B.1: The flow chart of the authentication process.

B.1 Performance analysis

We have tested the mobile applications on Blackberry Bold 9000 (BB9000) (4 devices), Blackberry Storm 9500 (BB9500) (1 device), HTC Wildfire (HTC) (1 device), Dell Streak (Dell) (1 device), iPhone 3 (1 device), iPad 1 (2 devices). 10 volunteers joined this test. They were located at different addresses. Coordination was made via phone calls, sending SMSs, and messaging on OSNs. Note in order to simplify our test, the member-list was imported from a Facebook event. We assumed there was a trustworthy leader. Therefore, the semi-SHCBK protocol was used. A total of 20 messages are exchanged. The size of the data sent by one device is about 18 KBytes. Compared with using traditional public key certificates, our method allows binding of contextual data (e.g. photos, voices or videos) to the uncertified public key we use in addition to names. We call these secondary security information which can be used to improve security as well as usability. Figure B.3 shows the time consumption of bootstrapping a group of all the devices we have. The total time cost is around 193 seconds.

We can see the cost of coordination is high in group formation because of many uncontrolled random factors. However, the verification and comparison is efficient and only takes a small fraction of the total time.

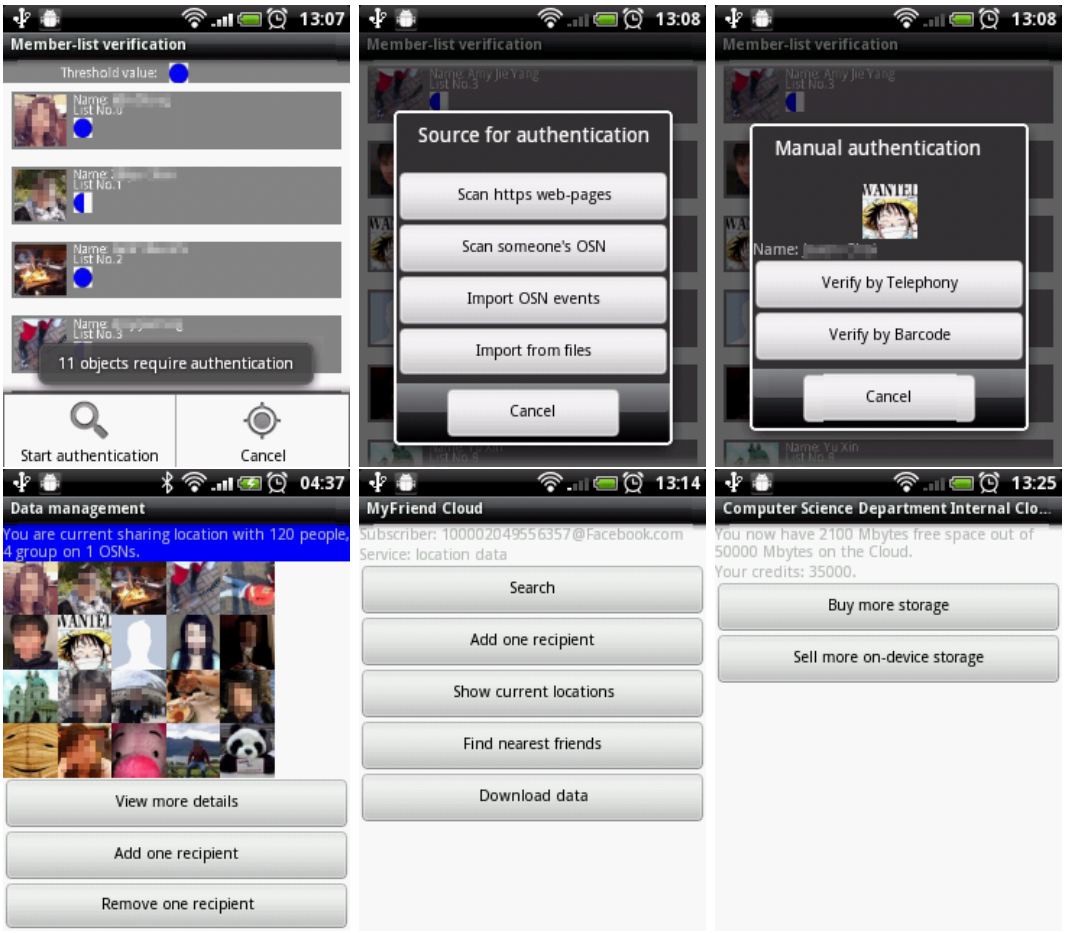


Figure B.2: Screen shots of the mobile application.

Device	Time	Ratio	Speed1	Speed2
BB9000	3.69s	99%	1.72kb/s	4.32kb/s
BB9500	4.49s	99%	1.35kb/s	3.75kb/s
HTC	3.74s	99%	1.56kb/s	4.80kb/s
Dell	0.85s	99%	2.42kb/s	7.15kb/s
iPhone	0.11s	99%	4.38kb/s	8.74kb/s
iPad	0.08s	99%	4.06kb/s	13.7kb/s

Table B.1: Facts and statistics.

Table B.1 shows the facts and statistics of different devices. The second column is the time of computing DH secret; the third column is the ratio of the time of computing DH secret against the time of total on-device computing (excluding communication); the fourth column is the speed of connection between the device and the coordination server; the last column is the speed of the connection between the

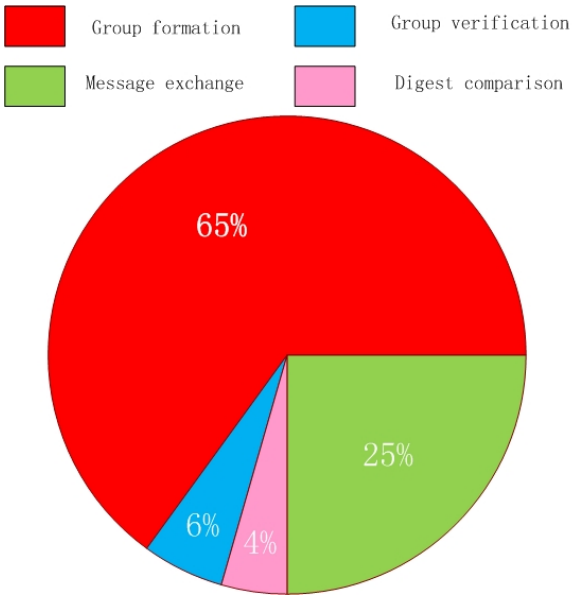


Figure B.3: Time consumption.

device and the Facebook server. We can see the time of on-device computation mostly originates from the DH secret computation.

According to the above analysis, we can identify two challenges for the future: (A) providing more convenient methods for large ad hoc group formation; (B) increasing the speed of mobile connections to allow including more contextual data in the protocol. Challenge A requires research on both security and usability. For example, should a group be formed using a single initiator, a tree structure, broadcasting over a fully connected graph, or some other topology? Challenge B is less significant since there are continuous developments in improving the speed of mobile connections; for example, the deployment of 4G networks.

Appendix C

List of Acronyms

ALE Annualised Loss Expectancy

CAP Card Authentication Program

CVSS Common Vulnerability Scoring System

CA Central Authority

DH Diffi-Hellman

DNS Domain Name Server

EMV Europay, MasterCard and VISA

GPS Global Positioning System

GPRS General Packet Radio Service

HCBK Hash Commitment Before Knowledge

HISP Human Interactive Security Protocol

HP Healthcare Provider

iKP Internet Keyed Payment

LED Light-emitting diode

MITM Man in the Middle

MITS Man in the Shop or Man in the Street

MITB Man in the Browser

MMMS Mobile Medical Monitoring System

MSD medical server domain

NFC Near Field Communication

NIST National Institute of Standard and Technology

OOB Out of Band

OSN Online Social Network

OBS On Body Sensor

OCTAVE Operationally Critical Threat, Asset, and Vulnerability Evaluation

PCI DSS Payment Card Industry Data Security Standard

PAD Patient Area Domain

PKI Public Key Infrastructure

SET Secure Electronic Transaction

SSL Secure Socket Layer

SAS Short Authentication String

TD Trusted Device

SHCBK Symmetric HCBK

SDL Security Development Lifecycle

3DS 3-D secure protocol

Bibliography

[1] Card Fraud Facts and Figures. http://www.ukpayments.org.uk/resources_publications/key_facts_and_figures/card_fraud_facts_and_figures accessed 30 March 2012.

[2] Fake DigiNotar web certificate risk to Iranians. http://www.bbc.co.uk/news/technology-14789763 accessed 10 November 2011.

[3] Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent. http://www.gartner.com/it/page.jsp?id=1848514 accessed 30 March 2012.

[4] Man-in-the-browser Trojan Attacks: Threat Analysis and Mitigation for Banks. Available on http://www.rsa.com/content_library.aspx?type_id=22.

[5] Map of cas. https://www.eff.org/files/colour_map_of_CAs.pdf accessed 30 March 2012.

[6] NFC Mobile Phone List. http://www.nfcworld.com/nfc-phones-list/ accessed 30 March 2012.

[7] NFC Posters. http://www.smart-poster.co.uk/nfc-smart-poster accessed 30 March 2012.

[8] NFC Sticker. http://www.engadget.com/2011/07/29/iphone-4-gets-upgraded-for-nfc-payments-the-hard-way/ accessed 30 March 2012.

[9] Oyster card. http://en.wikipedia.org/wiki/Oyster_card accessed 15 May 2011.

[10] Payment Card Industry (PCI) Data Security Standard. Available on https://www.pcisecuritystandards.org/security_standards/documents.php.

[11] Payments council to keep cheques and cancels 2018 target. http://www.paymentscouncil.org.uk/media_centre/press_releases/-/page/1575/.

[12] Qhost trojan profile. http://www.securelist.com/en/analysis/204792222/Mobile_Malware_Evolution_Part_5 accessed 12 August 2011.

[13] Société générale online payment description. https://particuliers.societegenerale.fr/votre_site/configuration_securite/paiements_securises.html accessed 10 November 2011.

[14] Special Report, Quarter 4, 2010. http://eumvno.files.wordpress.com/2010/10/bluetooth2010q4-dl.pdf accessed 30 August 2011.

[15] The New PayPal iPhone App. http://www.youtube.com/watch?v=suCe4-SWsHo.

[16] USB Phone Prices. http://www.von-phone.com/skype-phones.php accessed 5 September 2012.

[17] Videos of implementations of using HISPs. http://www.cs.ox.ac.uk/hcbk.

[18] When Hackers Attack: Six Companies Have Fallen Victim to Major Cyber Attacks in Recent Months. http://www.washingtonpost.com/business/economy/when-hackers-attack/2011/06/09/AGJMEBOH_gallery.html accessed 10 August 2011.

[19] Wifi direct api. http://developer.android.com/guide/topics/wireless/wifip2p.html accessed 10 January 2012.

[20] FIPS PUB 140-2: Security Requirements for Cryptography Modules, 2001. csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf.

[21] SmartMX Datasheet, 2008. http://www.nxp.com/documents/data_sheet/P5CX012_02X_40_73_80_144_FAM_SDS.pdf.

[22] HSBC deploys Authentify Out-of-band Authentication System, 2009. http://www.finextra.com/news/announcement.aspx?pressreleaseid=25285 accessed 15 May 2011.

[23] China Telecom, Bank of China and China UnionPay Launch Mobile Proximity Payments, 2010. http://www.finextra.com/news/announcement.aspx?pressreleaseid=36776 accessed 15 May 2011.

[24] Dutch deal paves way for mobile payments in 2012, 2010. http://uk.reuters.com/article/idUKLDE6880OC20100909 accessed 15 May 2011.

[25] ITU Sees 5 Billion Mobile Subscriptions Globally in 2010, 2010. http://www.itu.int/net/pressoffice/press_releases/2010/06.aspx accessed 15 May 2011.

[26] NXP Selected to Secure New German National Identity Card, 2010. http://www.nxp.com/news/content/file_1750.html accessed 15 May 2011.

[27] Personal Computers market, 2010. http://www.areppim.com/stats/stats_pcxfcst.htm accessed 15 May 2011.

[28] iZettle - Payments On The Go, 2011. http://www.youtube.com/watch?v=8GMrIxi2rc0 accessed 5 July 2012.

[29] Aloul, F., Zahidi, S., and El-Hajj, W. Multi factor authentication using mobile phones. *Mathematics and Computer Science 4*, 2 (2009), 65–80.

[30] Anderson, R. Position Statement in RFID S&P Panel: RFID and the Middleman. In *Financial Cryptography and Data Security*, S. Dietrich and R. Dhamija, Eds., vol. 4886 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2007, pp. 46–49.

[31] Anderson, R. J. *Security Engineering: A Guide to Building Dependable Distributed Systems*, 2 ed. Wiley Publishing, 2008.

[32] Asokan, N., Niemi, V., and Nyberg, K. Man-in-the-Middle in Tunnelled Authentication Protocols. In *Security Protocols*, B. Christianson, B. Crispo, J. Malcolm, and M. Roe, Eds., vol. 3364 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2005, pp. 28–41. 10.1007/11542322_6.

[33] Backhouse, J., Hsu, C., and McDonnell, A. Toward public-key infrastructure interoperability. *Commun. ACM 46*, 6 (June 2003), 98–100.

[34] Baker, W., Goudie, M., Hutton, A., Hylender, C. D., Niemantsverdriet, J., Novak, C., Ostertag, D., Porter, C., Rosen, M., Sartin, B., and Peter Tippett, M. 2011 Data Breach Investigations Report. `http://www.verizonbusiness.com/resources/reports/rp_data-breach-investigations-report-2011_en_xg.pdf` accessed 3 June 2012.

[35] Bakhtiari, S., Baraani, A., and Khayyambashi, M.-R. MobiCash: A New Anonymous Mobile Payment System Implemented by Elliptic Curve Cryptography. In *Computer Science and Information Engineering, 2009 WRI World Congress on* (31 2009-april 2 2009), vol. 3, pp. 286 –290.

[36] Balan, R. K., Ramasubbu, N., Prakobphol, K., Christin, N., and Hong, J. mFerio: The Design and Evaluation of A Peer-to-peer Mobile Payment System. In *Proceedings of the 7th international conference on Mobile systems, applications, and services* (New York, NY, USA, 2009), MobiSys '09, ACM, pp. 291–304.

[37] Balfanz, D., Smetters, D. K., Stewart, P., and Wong, H. C. Talking to strangers: Authentication in ad-hoc wireless networks. In *In Symposium on Network and Distributed Systems Security (NDSS '02), San Diego, California* (2002).

[38] Barker, E., Barker, W., Burr, W., Polk, W., and Smid, M. NIST Special Publication 800-57: Recommendation for Key Management Part 1: General(Revised), 2007. `http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf`.

[39] Barroso, D. ZeuS Mitmo: Man-in-the-mobile. `http://securityblog.s21sec.com/2010/09/zeus-mitmo-man-in-mobile-i.html` accessed 10 November 2011.

[40] Becher, M., Freiling, F., Hoffmann, J., Holz, T., Uellenbeck, S., and Wolf, C. Mobile Security Catching Up? Revealing the Nuts and Bolts of the Security of Mobile Devices. In *Security and Privacy (SP), 2011 IEEE Symposium on* (may 2011), pp. 96 –111.

[41] Bella, G., Paulson, L. C., and Massacci, F. The Verification of an Industrial Payment Protocol: the SET Purchase Phase. In *Proceedings of the 9th ACM conference on Computer and communications security* (New York, NY, USA, 2002), CCS '02, ACM, pp. 12–20.

[42] Bellare, M., Garay, J. A., Hauser, R., Herzberg, A., Krawczyk, H., Steiner, M., Tsudik, G., and Waidner, M. iKP: a Family of Secure Electronic Payment Protocols. In *Proceedings of the 1st conference on USENIX Workshop on Electronic Commerce - Volume 1* (Berkeley, CA, USA, 1995), USENIX Association, pp. 7–7.

[43] Buyens, K., De Win, B., and Joosen, W. Empirical and Statistical Analysis of Risk Analysis-driven Techniques for Threat Management. In *Proceedings of the The Second International Conference on Availability, Reliability and Security* (Washington, DC, USA, 2007), ARES '07, IEEE Computer Society, pp. 1034–1041.

[44] Cagalj, M., Saxena, N., and Uzun, E. On the Usability of Secure Association of Wireless Devices Based on Distance Bounding. In *Cryptology and Network Security* (2009).

[45] Callegati, F., Cerroni, W., and Ramilli, M. Man-in-the-Middle Attack to the HTTPS Protocol. *Security Privacy, IEEE 7*, 1 (jan.-feb. 2009), 78 –81.

[46] Capkun, S., and Cagalj, M. Integrity regions: authentication through presence in wireless networks. In *WiSe '06: Proceedings of the 5th ACM workshop on Wireless security* (2006).

[47] Card, S. K., Newell, A., and Moran, T. P. *The Psychology of Human-Computer Interaction*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1983.

[48] Cellan-Jones, R. Mobile money - has its moment come? `http://www.bbc.co.uk/news/technology-17057570` accessed 16 February 2012.

[49] Chen, B., Nguyen, L. H., and Roscoe, A. W. Reverse Authentication in Financial Transactions and Identity Management. *Mobile Networks and Applications*, 1–16. 10.1007/s11036-012-0366-2.

[50] Chen, B., Nguyen, L. H., and Roscoe, A. W. When Context Is Better Than Identity: Authentication by Context Using Empirical Channels. In *Security Protocols Workshop* (2011), pp. 115–125.

[51] Chen, B., and Roscoe, A. Social Networks for Importing and Exporting Security. In *To appear in 17th Monterey Workshop on Development, Operation and Management of Large-Scale Complex IT Systems* (2012).

[52] Chen, C.-H. O., Chen, C.-W., Kuo, C., Lai, Y.-H., McCune, J. M., Studer, A., Perrig, A., Yang, B.-Y., and Wu, T.-C. GAnGS: Gather, Authenticate 'n Group Securely. In *Proceedings of the 14th ACM international conference on Mobile computing and networking* (New York, NY, USA, 2008), MobiCom '08, ACM, pp. 92–103.

[53] Chen, J. J., and Adams, C. Short-range wireless technologies with mobile payments systems. In *Proceedings of the 6th international conference on Electronic commerce* (New York, NY, USA, 2004), ICEC '04, ACM, pp. 649–656.

[54] Clark, S. NFC Parking Meters. `http://www.nfcworld.com/2011/12/18/311965/san-francisco-gets-nfc-parking-meters/` accessed 30 March 2012.

[55] Courtney, Jr., R. H. Security Risk Assessment in Electronic Data Processing Systems. In *Proceedings of the June 13-16, 1977, national computer conference* (New York, NY, USA, 1977), AFIPS '77, ACM, pp. 97–104.

[56] Cukier, W. L., Cody, S., and Nesselroth, E. J. Genres of Spam: Expectations and Deceptions. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences - Volume 03* (Washington, DC, USA, 2006), IEEE Computer Society, pp. 51.1–.

[57] Dahlberg, T., Mallat, N., Ondrus, J., and Zmijewska, A. Past, Present and Future of Mobile Payments Research: A Literature Review. *Electronic Commerce Research and Applications 7*, 2 (2008), 165 – 181. Special Section: Research Advances for the Mobile Payments Arena.

[58] Dey, A. K. Understanding and Using Context. *Personal and Ubiquitous Computing 5* (2001), 4–7. 10.1007/s007790170019.

[59] Dolan, J. Accelerating the Development of Mobile Money Ecosystems. In *MOBILE MONEY SUMMIT 2009* (2009).

[60] Douceur, J. The Sybil Attack. In *Peer-to-Peer Systems*, vol. 2429 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2002, pp. 251–260.

[61] Drimer, S., Murdoch, S., and Anderson, R. Optimised to Fail: Card Readers for Online Banking. In *Financial Cryptography and Data Security*, R. Dingledine and P. Golle, Eds., vol. 5628 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2009, pp. 184–200. 10.1007/978-3-642-03549-4_11.

[62] Eckersley, P., and Burns, J. The Decentralized, 2011. `http://www.usenix.org/events/sec11/tech/slides/eckersley.pdf`.

[63] Ellison, C., and Schneier, B. Ten Risks of PKI: What You're Not Being Told About Public Key Infrastructure. In *Computer Security Journal* (2000). Available online at `http://www.schneier.com/paper-pki.html`.

[64] Felt, A. P., Finifter, M., Chin, E., Hanna, S., and Wagner, D. A Survey of Mobile Malware in the Wild. In *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices* (New York, NY, USA, 2011), SPSM '11, ACM, pp. 3–14.

[65] Fleizach, C., Liljenstam, M., Johansson, P., Voelker, G. M., and Mehes, A. Can You Infect Me Now?: Malware Propagation in Mobile Phone Networks. In *Proceedings of the 2007 ACM workshop on Recurring malcode* (New York, NY, USA, 2007), WORM '07, ACM, pp. 61–68.

[66] Francis, L., Hancke, G., Mayes, K., and Markantonakis, K. Practical NFC Peer-to-Peer Relay Attack Using Mobile Phones. In *Radio Frequency Identification: Security and Privacy Issues*, S. Ors Yalcin, Ed., vol. 6370 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2010, pp. 35–49. 10.1007/978-3-642-16822-2_4.

[67] Gao, J., Edunuru, K., Cai, J., and Shim, S. P2P-Paid: A Peer-to-Peer Wireless Payment System. In *Proc. Second IEEE International Workshop on Mobile Commerce and Services WMCS '05* (19–19 July 2005), pp. 102–111.

[68] Gehrmann, C., Mitchell, C. J., and Nyberg, K. Manual authentication for wireless devices. *RSA Cryptobytes 7* (2004).

[69] Golbeck, J., and Hendler, J. Accuracy of Metrics for Inferring Trust and Reputation in Semantic Web-Based Social Networks. In *Engineering Knowledge in the Age of the Semantic Web*, E. Motta, N. Shadbolt, A. Stutt, and N. Gibbins, Eds., vol. 3257 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2004, pp. 116–131. 10.1007/978-3-540-30202-5_8.

[70] Goodrich, M., Sirivianos, M., Solis, J., Tsudik, G., and Uzun, E. Loud and Clear: Human-Verifiable Authentication Based on Audio. In *Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on* (2006), p. 10.

[71] Granger, S. Social Engineering Fundamentals, Part I: Hacker Tactics. In *SecurityFocus* (2001).

[72] Guo, Z., Okuyama, T., and Finley, M. A new trust model for pki interoperability. In *Autonomic and Autonomous Systems and International Conference on Networking and Services, 2005. ICAS-ICNS 2005. Joint International Conference on* (2005), pp. 37–37.

[73] Hancke, G. A Practical Relay Attack on ISO 14443 Proximity Cards., 2005. `http://www.rfidblog.org.uk/hancke-rfidrelay.pdf`.

[74] Hancke, G. P. Practical Attacks on Proximity Identification Systems. In *Proceedings of IEEE Symposium on Security and Privacy* (2006), pp. 328–333.

[75] Harris, B., and Hunt, R. TCP/IP Security Threats and Attack Methods. *Computer Communications 22*, 10 (1999), 885 – 897.

[76] Haselsteiner, E., and Breitfuss, K. Security in Near Field Communication. In *Workshop on RFID Security* (2006).

[77] Hassinen, M., Hyppnen, K., and Haataja, K. An Open, PKI-Based Mobile Payment System. In *Emerging Trends in Information and Communication Security*, G. Mller, Ed., vol. 3995 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2006, pp. 86–100. 10.1007/11766155_7.

[78] Hastings, N., and McLean, P. TCP/IP Spoofing Fundamentals. In *Computers and Communications, 1996., Conference Proceedings of the 1996 IEEE Fifteenth Annual International Phoenix Conference on* (mar 1996), pp. 218 – 224.

[79] Herley, C., van Oorschot, P., and Patrick, A. Passwords: If Were So Smart, Why Are We Still Using Them? In *Financial Cryptography and Data Security*, R. Dingledine and P. Golle, Eds., vol. 5628 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2009, pp. 230–237. 10.1007/978-3-642-03549-4_14.

[80] Huang, X., Chen, B., Markham, A., Wang, Q., Zheng, Y., and Roscoe, A. W. Human Interactive Secure Key and ID Exchange Protocols in Body Sensor Networks. In *Accepted by IET Information Security, Special Issue on Trust and Identity Management in Mobile and Internet Computing and Communications* (2013).

[81] Huang, X., Ma, X., Chen, B., Markham, A., Wang, Q., and Roscoe, A. W. Human Interactive Secure ID Management in Body Sensor Network. *Accepted by Journal of Networks* (2012).

[82] Huang, X., Wang, Q., Bangdao, C., Markham, A., Jäntti, R., and Roscoe, A. W. Body Sensor Network Key Distribution Using Human Interactive Channels. In *Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies* (New York, NY, USA, 2011), ISABEL '11, ACM, pp. 143:1–143:5.

[83] Hwang, R.-J., Shiau, S.-H., and Jan, D.-F. A New Mobile Payment Scheme for Roaming Services. *Electron. Commer. Rec. Appl. 6*, 2 (July 2007), 184–191.

[84] Jakobsson, M., and Wetzel, S. Security Weaknesses in Bluetooth. In *Lecture Notes in Computer Science* (2001), vol. 2020, pp. 176+.

[85] Jansen, W., and Scarfone, K. Nist special publication 800-124: Guidelines on cell phone and pda security, 2008. `csrc.nist.gov/publications/nistpubs/800-124/SP800-124.pdf` accessed 20 May 2011.

[86] Johnson, M. A new approach to Internet banking. Tech. Rep. UCAM-CL-TR-731, University of Cambridge, Computer Laboratory, 09 2008.

[87] Johnson, M., and Moore, S. A New Approach to E-Banking. In *Proc. 12th Nordic Workshop on Secure IT Systems (NORDSEC 2007)* (Oct 2007), Úlfar Erlingsson and A. Sabelfeld, Eds., University of Reykjavik, pp. 127–138. `http://www.matthew.ath.cx/publications/2007-Johnson-ebanking.pdf`.

[88] Kadambi, K. S., Li, J., and Karp, A. H. Near-field Communication-based Secure Mobile Payment Service. In *Proceedings of the 11th International Conference on Electronic Commerce* (New York, NY, USA, 2009), ICEC '09, ACM, pp. 142–151.

[89] Kainda, R., Flechais, I., and Roscoe, A. *Information Security Theory and Practice. Security and Privacy of Pervasive Systems and Smart Devices*, vol. 6033 of *WISTP 2010, Lecture Notes in Computer Sciences*. Spinger, 4 2010, ch. Secure and Usable Out-Of-Band Channels for Ad hoc Mobile Device Interactions, pp. 308–315.

[90] Kainda, R., Flechais, I., and Roscoe, A. W. Usability and Security of Out-Of-Band Channels in Secure Device Pairing Protocols. In *Proceedings of the 5th Symposium on Usable Privacy and Security* (New York, NY, USA, 2009), SOUPS '09, ACM, pp. 11:1–11:12.

[91] Karger, P. A., Toll, D. C., Palmer, E. R., McIntosh, S. K., Weber, S., and Edwards, J. W. Implementing a High-Assurance Smart-Card OS. In *Proceeding of Financial Cryptography and Data Security* (2010), Springer.

[92] Karp, G. Suspicious Swipe Pads Removed From 80 Stores. `http://articles.chicagotribune.com/2011-05-11/business/ct-biz-0512-michaels-20110511_1_debit-card-swipe-pin-pads` accessed 10 November 2011.

[93] Kindberg, T., Zhang, K., and Shankar, N. Context Authentication Using Constrained Channels. In *Mobile Computing Systems and Applications, 2002. Proceedings Fourth IEEE Workshop on* (2002), pp. 14 – 21.

[94] Krebs, B. All About Skimmers. `http://krebsonsecurity.com/all-about-skimmers/` accessed 20 September 2012.

[95] Kumparak, G. Android's new face unlock feature can be fooled with a photo. http://techcrunch.com/2011/11/11/android-facial-unlock-photo/ accessed 8 February 2012.

[96] Kgler, D. man in the middle attacks on bluetooth. In *Computer Aided Verification*, W. Hunt and F. Somenzi, Eds., vol. 2742 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2003, pp. 149–161. 10.1007/978-3-540-45126-6_11.

[97] Laur, S., and Nyberg, K. Efficient Mutual Data Authentication Using Manually Authenticated Strings. In *Proceeding of Cryptology and Network Security* (2006), Springer, pp. 90–107.

[98] Lawton, G. Is It Finally Time to Worry about Mobile Malware? *Computer 41*, 5 (may 2008), 12 –14.

[99] Lin, Y.-H., Studer, A., Hsiao, H.-C., McCune, J. M., Wang, K.-H., Krohn, M., Lin, P.-L., Perrig, A., Sun, H.-M., and Yang, B.-Y. SPATE: Small-group PKI-less Authenticated Trust Establishment. In *Proceedings of the 7th international conference on Mobile systems, applications, and services* (New York, NY, USA, 2009), MobiSys '09, ACM, pp. 1–14.

[100] Lindell, A. Comparison-Based Key Exchange and the Security of the Numeric Comparison Mode in Bluetooth v2.1. In *Topics in Cryptology CT-RSA 2009*, M. Fischlin, Ed., vol. 5473 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2009, pp. 66–83. 10.1007/978-3-642-00862-7_5.

[101] Lutz, Z. NFC SIM Cards. http://www.engadget.com/2011/11/17/sim-based-nfc-gains-global-support-from-45-mobile-carriers-all/ accessed 30 March 2012.

[102] Mallat, N. Exploring consumer adoption of mobile payments - a qualitative study. *The Journal of Strategic Information Systems 16*, 4 (2007), 413–432.

[103] Mannan, M., and van Oorschot, P. Using a Personal Device to Strengthen Password Authentication from an Untrusted Computer. In *Financial Cryptography and Data Security*, S. Dietrich and R. Dhamija, Eds., vol. 4886 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2007, pp. 88–103. 10.1007/978-3-540-77366-5_11.

[104] Maslennikov, D. Mobile Malware Evolution, Part 5. `http://www.securelist.com/en/analysis/204792222/Mobile_Malware_Evolution_Part_5` accessed 20 March 2012.

[105] Mathieu, and Gorge. Data Protection: Why Are Organisations Still Missing the Point? *Computer Fraud and Security 2008*, 6 (2008), 5 – 8.

[106] Mayrhofer, R., and Gellersen, H. Shake well before use: Authentication based on Accelerometer Data. In *Proc. Pervasive 2007: 5th International Conference on Pervasive Computing* (May 2007), vol. 4480 of *LNCS*, Springer-Verlag, pp. 144–161.

[107] McCune, J., Perrig, A., and Reiter, M. Seeing-is-believing: Using Camera Phones for Human-verifiable Authentication. In *Security and Privacy, 2005 IEEE Symposium on* (may 2005), pp. 110 – 124.

[108] Meier, J., Mackman, A., Dunner, M., Vasireddy, S., Escamilla, R., and Murukan, A. Improving Web Application Security: Threats and Countermeasures: Chapter 3 Threat Modeling. Available on `http://msdn.microsoft.com/en-us/library/ff648644.aspx` accessed 10 Oct 2011.

[109] Menezes, A. J., van Oorschot, P. C., and Vanstone, S. A. *Handbook of Applied Cryptography*. John Wiley, 1997.

[110] Messmer, E. Fake GSM Base Station Trick Targets IPhones. `http://www.networkworld.com/news/2011/011911-black-hat-trick-iphones.html` accessed 30 August 2011.

[111] Michelle Keeney, J., Kowalski, E., Cappelli, D., Moore, A., Shimeall, T., and Rogers, S. Insider threat study: Computer system sabotage in critical infrastructure sectors. Report from the CERT Insider Threat Center, available on www.cert.org/archive/pdf/bankfin040820.pdfaccessed20August2011.

[112] Miller, B., Nixon, T., Tai, C., and Wood, M. Home networking with Universal Plug and Play. *Communications Magazine, IEEE 39*, 12 (dec 2001), 104 –109.

[113] Mills, E. Researchers find avenues for fraud in Square, 2011. http://news.cnet.com/8301-27080_3-20088441-245/researchers-find-avenues-for-fraud-in-square/ accessed 30 August 2011.

[114] Mune, C., Gassira, R., and Piccirillo, R. Hijacking Mobile Data Connections, 2008. http://www.blackhat.com/presentations/bh-europe-09/Gassira_Piccirillo/BlackHat-Europe-2009-Gassira-Piccirillo-Hijacking-Mobile-Data-Connections-whitepaper.pdf accessed 15 May 2011.

[115] Murdoch, S., and Anderson, R. Verified by Visa and MasterCard SecureCode: Or, How Not to Design Authentication. In *Financial Cryptography and Data Security*, R. Sion, Ed., vol. 6052 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2010, pp. 336–342. 10.1007/978-3-642-14577-3_27.

[116] Murdoch, S. J., Drimer, S., Anderson, R., and Bond, M. Chip and pin is broken. In *Security and Privacy (SP), 2010 IEEE Symposium on* (may 2010), pp. 433 –446.

[117] Namestnikov, Y. Kaspersky Security Bulletin. Statistics 2011. http://www.securelist.com/en/analysis/204792216/Kaspersky_Security_Bulletin_Statistics_2011 accessed 20 March 2012.

[118] Namestnikov, Y. It threat evolution: Q3 2011, 2011. http://www.securelist.com/en/analysis/204792201/IT_Threat_Evolution_Q3_2011 accessed 15 December 2011.

[119] Nguyen, L. H., Ed. ch. Part 6: Mechanisms using manual data transfer.

[120] Nguyen, L. H., and Roscoe, A. W. Efficient Group Authentication Protocol Based on Human Interaction. In *Proceeding of Workshop on Foundation of Computer Security and Automated Reasoning Protocol Security Analysis* (2006), pp. 9–31.

[121] Nguyen, L. H., and Roscoe, A. W. Authenticating Ad Hoc Networks by Comparison of Short Digests. *Information and Computation 206* (2008), 250–271.

[122] Nguyen, L. H., and Roscoe, A. W. Separating two roles of hashing in one-way message authentication. In *FCS-ARSPA-WITS* (2008).

[123] Nguyen, L. H., and Roscoe, A. W. Authentication Protocols Based on Low-bandwidth Unspoofable Channels: A Comparative Survey. *J. Comput. Secur. 19*, 1 (Jan. 2011), 139–201.

[124] Nguyen, L. H., and Roscoe, A. W. Short-output Universal Hash Functions and Their Use in Fast and Secure Message Authentication. *FSE 2012 to appear* (2012).

[125] Ølnes, J. Pki interoperability by an independent, trusted validation authority. In *5th Annual PKI R&D Workshop, NIST Gaithersburg MD, USA* (2006).

[126] Ornaghi, A., and Valleri, M. Man in the middle attacks. http://www.blackhat.com/presentations/bh-europe-03/bh-europe-03-valleri.pdf.

[127] Parno, B., Kuo, C., and Perrig, A. Phoolproof Phishing Prevention. In *Financial Cryptography and Data Security*, G. Di Crescenzo and A. Rubin, Eds.,

vol. 4107 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2006, pp. 1–19. 10.1007/11889663_1.

[128] Pasquet, M., Reynaud, J., and Rosenberger, C. Secure payment with nfc mobile phone in the smarttouch project. In *Proceeding of International Symposium on Collaborative Technologies and Systems* (2008).

[129] Pasquet, M., Reynaud, J., and Rosenberger, C. Secure Payment with NFC Mobile Phone in the SmartTouch Project. In *Collaborative Technologies and Systems, 2008. CTS 2008. International Symposium on* (may 2008), pp. 121 –126.

[130] Polk, W. T., Dodson, D. F., Burr, W. E., Ferraiolo, H., and Cooper, D. NIST Special Publication 800-78-3: Cryptographic Algorithms and Key Sizes for Personal Identity Verification, 2010. `http://csrc.nist.gov/publications/nistpubs/800-78-3/sp800-78-3.pdf`.

[131] Poulsen, K. Credit Card Skimming Survey: Whats Your Magstripe Worth? `http://www.wired.com/threatlevel/2009/10/florida_skimming/` accessed 10 November 2011.

[132] Poulsen, K. Washington D.C. Restaurants Become Credit Card Cloning Hot Spots. `http://www.wired.com/threatlevel/2009/03/washington/` accessed 10 November 2011.

[133] Pradhan, S., Lawrence, E., and Zmijewska, A. Bluetooth as An Enabling Technology in Mobile Transactions. In *Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on* (april 2005), vol. 2, pp. 53 – 58 Vol. 2.

[134] Rainer, Jr., R. K., Snyder, C. A., and Carr, H. H. Risk Analysis for Information Technology. *J. Manage. Inf. Syst. 8*, 1 (June 1991), 129–147.

[135] Rao, L. VeriFone Takes The Gloves Off, Accuses Square Of Serious Security Hole, 2011. `http://techcrunch.com/2011/03/09/verifone-takes-the-`

gloves-off-accuses-square-of-serious-security-hole/ accessed 30 August 2011.

[136] Rice, A. A Continued Commitment to Security. http://www.facebook.com/blog/blog.php?post=486790652130 accessed 30 August 2011.

[137] Roscoe, A., Smyth, T., and Nguyen, L. Model checking cryptographic protocols subject to combinatorial attack. Avalable on http://www.cs.ox.ac.uk/files/4157/guess.pdf.

[138] Roscoe, A. W. Human-centred computer security. Unpublished draft, available on http://web.comlab.ox.ac.uk/oucl/work/bill.roscoe/publications/113.pdf, 2006.

[139] Sadeghi, A.-R., and Schneider, M. Electronic Payment Systems. In *Digital Rights Management*, E. Becker, W. Buhse, D. Gnnewig, and N. Rump, Eds., vol. 2770 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2003, pp. 113–137.

[140] Saxena, A., Das, M. L., and Gupta, A. MMPS: A Versatile Mobile-to-mobile Payment System. In *Mobile Business, 2005. ICMB 2005. International Conference on* (july 2005), pp. 400 – 405.

[141] Saxena, N., Ekberg, J.-E., Kostiainen, K., and Asokan, N. Secure device pairing based on a visual channel. In *Security and Privacy, 2006 IEEE Symposium on* (may 2006), pp. 6 pp. –313.

[142] Saxena, N., and Uddin, M. Blink em all: Scalable, user-friendly and secure initialization of wireless sensor nodes. In *Cryptology and Network Security*, J. Garay, A. Miyaji, and A. Otsuka, Eds., vol. 5888 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2009, pp. 154–173. 10.1007/978-3-642-10433-6_11.

[143] Selén, K. UPnP Security in Internet Gateway Devices. In *Proceedings of Telecommunications Software and Multimedia, Espoo, Finland* (2006).

[144] Silverman, R. E. Dsniff and SSH : Reports of My Demise are Greatly Exaggerated. http://www.oreillynet.com/pub/a/oreilly/networking/news/silverman_1200.html accessed 15 May 2011.

[145] Soriente, C., Tsudik, G., and Uzun, E. HAPADEP: Human-Assisted Pure Audio Device Pairing. In *Information Security*, T.-C. Wu, C.-L. Lei, V. Rijmen, and D.-T. Lee, Eds., vol. 5222 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2008, pp. 385–400. 10.1007/978-3-540-85886-7_27.

[146] Srivastava, L. Japan's ubiquitous mobile information society. *Policy, Regulation and Strategy for Telecommunications 6*, 4 (2004).

[147] Stajano, F. Pico: No More Passwords! In *Security Protocols XIX* (2011), B. Christianson, B. Crispo, J. Malcolm, and F. Stajano, Eds., vol. 7114 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 49–81.

[148] Stoneburner, G., Goguen, A., and Feringa, A. Nist special publication 800-30: Risk management guide for information technology systems, 2002. csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf accessed 20 March 2011.

[149] Struif, B. Use of Biometrics for User Verification in Electronic Signature Smartcards. In *Proceeding of International Conference on Research in Smart Cards: Smart Card Programming and Security* (2001).

[150] T. Holgers, D. W., and Gribble, S. Cutting Through the Confusion: A Measurement Study of Homograph Attacks. In *Proceedings of USENIX Annual Technical Conference (USENIX)* (2006).

[151] Thornburgh, T. Social Engineering: the "Dark Art". In *Proceedings of the 1st annual conference on Information security curriculum development* (New York, NY, USA, 2004), InfoSecCD '04, ACM, pp. 133–135.

[152] Tobin, D. Open sesame: the magic car thieves, 2011. http://www.thesundaytimes.co.uk/sto/ingear/cars/Driving/article533163.ece accessed 15 August 2011.

[153] Uzun, E., Karvonen, K., and Asokan, N. Usability analysis of secure pairing methods. In *Financial Cryptography and Data Security*, S. Dietrich and R. Dhamija, Eds., vol. 4886 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2007, pp. 307–324.

[154] Vaudenay, S. Secure Communications over Insecure Channels based on Short Authenticated Strings. In *Proceeding of Advances in Cryptology - Crypto* (2005), Springer, pp. 309–326.

[155] Vennon, T. Mobile Malware Development Continues To Rise, Android Leads The Way. http://forums.juniper.net/t5/Security-Mobility-Now/Mobile-Malware-Development-Continues-To-Rise-Android-Leads-The/ba-p/132695 accessed 15 January 2012.

[156] VISA, M. . Secure Electronic Transactions (SET) Specifications, 1997. Available on http://www.cl.cam.ac.uk/research/security/resources/SET/.

[157] Vishnani, K., Pais, A. R., and Mohandas, R. An In-Depth Analysis of the Epitome of Online Stealth: Keyloggers; and Their Countermeasures. In *ACC (3)* (2011), A. Abraham, J. L. Mauri, J. F. Buford, J. Suzuki, and S. M. Thampi, Eds., vol. 192 of *Communications in Computer and Information Science*, Springer, pp. 10–19.

[158] Wang, Y.-M., Beck, D., Wang, J., Verbowski, C., and Daniels, B. Strider Typo-patrol: Discovery and Analysis of Systematic Typo-squatting. In *Proceedings of the 2nd conference on Steps to Reducing Unwanted Traffic on the Internet - Volume 2* (Berkeley, CA, USA, 2006), USENIX Association, pp. 5–5.

[159] Weigold, T., Kramp, T., Hermann, R., Hring, F., Buhler, P., and Baentsch, M. The Zurich Trusted Information Channel An Efficient Defence Against Man-in-the-Middle and Malicious Software Attacks. In *Trusted Computing - Challenges and Applications*, P. Lipp, A.-R. Sadeghi, and K.-M. Koch, Eds., vol. 4968 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2008, pp. 75–91. 10.1007/978-3-540-68979-9_6.

[160] Zhu, Y., and Rice, J. A Lightweight Architecture for Secure Two-Party Mobile Payment. In *Computational Science and Engineering, 2009. CSE '09. International Conference on* (aug. 2009), vol. 2, pp. 326 –333.

[161] Zolfaghar, K., and Mohammadi, S. Securing Bluetooth-based Payment System Using Honeypot. In *Innovations in Information Technology, 2009. IIT '09. International Conference on* (dec. 2009), pp. 21 –25.