Available online at www.sciencedirect.com

ScienceDirect

Electronic Commerce Research and Applications 7 (2008) 214–231

Electronic Commerce Research and Applications

www.elsevier.com/locate/ecra

Utilizing national public-key infrastructure in mobile payment systems

Marko Hassinen [a,*], Konstantin Hyppönen [a], Elena Trichina [b]

[a] *University of Kuopio, Department of Computer Science, P.O.B. 1627, FI-70211, Kuopio, Finland*
[b] *Spansion International Inc., Willi-Brandt-Allee 4, 81829 Munich, Germany*

Received 18 September 2006; received in revised form 27 March 2007; accepted 28 March 2007
Available online 20 April 2007

Abstract

Payments are the locomotive behind any business domain. It has been predicted that mobile payments will become one of the most successful mobile services, and the security of payments is an important requirement. However, it is difficult to strongly authenticate mobile users remotely and provide an adequate level of non-repudiation of transactions. In this article, we argue that a nationwide public-key infrastructure supported by governmental bodies can be used in a mobile payment system. Not only does it provide strong security, but it also makes the system open to any mobile user, merchant, or financial service provider. Two payment protocols are described: one for virtual point-of-sale payments, and one for vending machine payments. We argue that such a system can be implemented using open development platforms, and its performance is adequate for enabling swift transactions. A prototype of a system which accepts virtual point-of-sale payments is implemented, and its performance and usability are evaluated.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Mobile payments; Wireless applications; PKI; Mobile ID

1. Introduction

With the flourishing of electronic commerce and widespread use of mobile devices, a new type of service is emerging that extends e-business using wireless technology by enabling e-commerce services on mobile devices. The pervasiveness of wireless networks and deployment of packet-switching technology create new opportunities for innovative mobile services such as dynamic location-based information (e.g., news, road conditions), entertainment (e.g., m-games, playback of video clips) and communication (e.g., mobile advertising, promotions). Great expectations are also being placed on mobile transaction services such as m-payments, m-banking, and m-auctions. The vision that mobile devices can be used as enablers for contactless proximity-based payments in traditional face-to-face commerce and peer-to-peer transactions is taking hold too.

The reasons for such high expectations are many. For users, mobile phones offer convenience, immediacy and personalization for consumer transactions. From the merchants' point of view, m-commerce allows enterprises to expand their market reach and reduce cost. From the mobile network operator's perspective, m-commerce has considerable potential in maintaining and increasing average revenue per user. Financial service providers (FSP) such as banks and credit card companies are seeing opportunities in using mobile phones as a personal (secure) payment terminal [1] as well as addressing a new customer base that traditionally operated in the sachet[1] economy [2].

It has been predicted that mobile payments have the potential to become one of the most important services in future mobile networks [3–5]. It is clear, however, that many issues have to be resolved before mass adoption

* Corresponding author. Tel.: +358 17 162559; fax: +358 17 162595.
E-mail addresses: Marko.Hassinen@uku.fi (M. Hassinen), Konstantin.Hypponen@uku.fi (K. Hyppönen), Elena.Trichina@spansion.com (E. Trichina).

[1] An economy where populace can only afford buying in small quantities and relies on cash for such transactions. While usually being outside of a banking system due to low income, the same group is known to be avid users of mobile phones with pre-paid top-up accounts.

1567-4223/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.elerap.2007.03.006

can occur; among them, security has a prominent role [6,7]. Indeed, influential industrial consortia such as Mobey Forum consider security to be a fundamental requirement for mobile payments and financial services to be valid and adopted by all stakeholders [1]. For customer acceptance, both technical and perceived levels of security should be high, so that customers do not suffer financial losses and their privacy is not compromised [8]. For businesses, effective customer authentication is cited as the most important element in facilitating mobile payments [9]. Transaction-level security must be end-to-end, with message integrity, confidentiality and non-repudiation guaranteed [10]. Non-repudiation (ensuring that a purchase contract cannot later be denied by any of the parties involved) is especially important in mobile payments. If non-repudiation is not ensured, users can potentially claim that the messages related to a transaction were generated by a rogue insider of the mobile network operator.

While architectural decisions concerning the level of security are left to service providers, strong authentication and non-repudiation based on wireless public-key infrastructure (WPKI) and digital certificates are required in macro-payments (i.e., payments higher than 10€) and in all mobile banking services [1,11].

This paper concentrates on the topic of secure mobile payments. Specifically, we describe an open PKI-based platform that facilitates the development of a wide range of secure mobile payment applications. It provides an open technological solution to secure mobile payment transactions that can be freely utilized by financial institutions, mobile network operators or independent third parties. Moreover, in line with the requirements of time to market, our solution relies on today's handset technology and is based on existing standards, such as SIM Application Toolkit (SAT) [12], and open certificate status protocol (OCSP) [13]. The proof-of-concept implementation is done entirely on open development tools and platforms such as Java 2 Micro Edition and Web Services technology [14].

The proposed solution allows us to provide strong customer authentication and non-repudiation by employing public-key cryptography for customer certificates and digital signatures. Confidentiality and message integrity are provided by encrypting messages that constitute mobile payment transactions. What sets our system apart is that it utilizes the existing national public-key infrastructure, which is independent of financial institutions, network operators and mobile payment intermediaries but can be used by all of them. Nowadays, such public-key infrastructure (PKI) is available in Finland, but in the near future it will become available within the whole of the EU [15] and in some countries of the Asia-Pacific region, as amply described in the "One card, one Asia" slogan, see http://www.asiaiccardforum.org. The use of a national PKI solves the basic problem of any system based on asymmetric cryptography, namely, who signs the root certificate and maintains the required infrastructure.

The rest of the paper is organized as follows. Before proceeding with a detailed description of our mobile payment platform, in the following section we review the context and concepts of mobile payment procedures as defined in various academic surveys [5,16–19] and white papers of industrial consortia [1,2,20]. We discuss payment risks, and then formulate the requirements of mobile payment transaction security and present different ways these requirements can be enabled using mobile phones. In Section 3 we give a short overview of the Finnish national public-key infrastructure, namely, FINEID. Next, we provide a specification of mobile payment protocols that guarantee strong authentication and non-repudiation. Section 4 describes protocols for two scenarios; one is for virtual point-of-sale (POS) payments and one is for contactless proximity-based payments at vending machines or physical POS. In Section 5 we give an example of a system that implements our virtual POS payment protocol. The proof-of-concept implementation is an application for buying train tickets using a mobile phone. The application architecture, technical details of the implementation, and test results are provided. We conclude the paper by discussing advantages and limitations of the proposed solution and possibilities of using the platform in other countries.

2. Mobile payment procedures and their security

Following [18], we define mobile payments as wireless transactions of monetary value from one party to another where a mobile device (e.g., mobile phone, PDA, smartphone or any mobile payment terminal) is used in order to initialize, activate and/or confirm the payment.

Mobile payments may be categorized into various types depending on a large number of parameters, such as the transaction settlement method (pre-paid, post-paid, or pay-now), purchase type (digital or real goods) and value (pico-, micro-, and macro-payments). Mobile payment environments can be divided into remote (mobilization of traditional e-commerce by enabling access to WAP shops, or virtual POS), local (by facilitating payment services at physical POS) and proximity (contactless variants of face-to-face commerce) payments. Various enabling (SIM, WIM, dual slot, external card reader) and interactive (voice, Short Message Service (SMS) or Unstructured Supplementary Service Data (USSD), WAP, RFID, Bluetooth) technologies are competing to become established standards for physical and virtual mobile payments. An encompassing summary of various mobile payment procedures and systems is given in [17,18,16] suggests an interesting basis for their systematic comparison in the form of a morphological box of mobile payment characteristics and instances.

2.1. Mobile payment framework

To introduce the mobile payment context, its players and their roles we use as a reference the enhanced variant of the model defined in [17]. The main actions within a mobile payment system are: registration with the service providers, initialization of payment transaction, consumer authentication, payment authorization and payment settlement. In the general case, these functions can be assigned to four principal actors of the payment system: customer, content provider/merchant, payment service provider (PSP), and trusted third party (TTP). The mobile payment model is depicted in Fig. 1.

The customer is a person who owns a mobile device and wants to use it to pay for content, goods or services. We often refer to a customer as a mobile user. The mobile user's roles may involve: (a) registering with a PSP; (b) initializing the mobile purchase; and (c) authorizing the payment. In the context of security, yet another, often overlooked, role is (d) identifying oneself as the legitimate owner of the mobile device.

The merchant's role may involve: (a) initialization of the purchase; (b) forwarding the purchase request to the PSP; (c) replaying authorization requests and responses between customers and payment service providers; and (d) delivery of goods/services/content.

The payment service provider, or acquirer, is the party responsible for the payment process. A mobile user may be required to register with a PSP (e.g., to open a pre-paid or credit/debit account with the PSP). A PSP can be a network operator, a bank, a credit card company or an independent payment vendor. One role of the PSP is to provide business rules and control transaction flow between the mobile user, the content provider/merchant and the trusted third party.

The trusted third party, or issuer, provides the service certificate for a given account. The account can be associated with monetary value or personal data and is usually only for services offered by the TTP. The service certificate is the means whereby the TTP identifies the mobile user, and varies from a PIN code to digital public-key certificates. The TTP can be a network operator, a bank, a credit card company or an intermediary. The role of the TTP is service certificate issuing, customer authentication and authorization of payment requests.

Mobile network operators, banks, and intermediaries may be positioned at the same time as the TTP, PSP

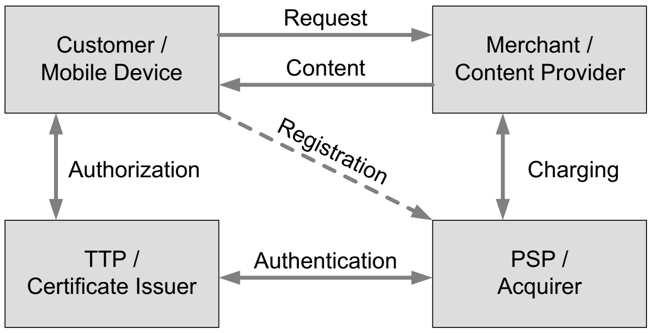


Fig. 1. Mobile payment model.

and/or content providers; therefore, a four-party model is often realized by two or three players.

2.2. Security features of mobile payment procedures: state of the art

In this section we portray typical current mobile payment systems and examine their security levels with respect to four main security functions, namely authentication (establishing customer's credentials), confidentiality (guarantee that transaction details are not disclosed to any unauthorized party), data integrity (assurance that payment data is not altered after the customer agrees to the terms of the transaction) and non-repudiation (binding parties to the transaction). This analysis is a stepping stone for describing various enabling technologies for secure mobile payments, their advantages and shortcomings. We conclude this section by outlining our answer to implementation challenges of PKI-based mobile payments security.

2.2.1. Typical mobile payment procedures and their security features

An example of a classical four-party mobile payment procedure is PayBox,[2] the best known payment system in Europe (http://www.paybox.net). This service is a payment intermediary not tied to any particular network or bank. It encompasses all payment scenarios and can be used for any payment of 0.01€ or higher; the transaction limit is 200€. In order to use the services, a mobile phone user has to register with PayBox providing her mobile phone ID (a phone number or alias) and bank account details; in return she receives a service-specific "PayBox-PIN" which is used to authorize the payment transactions; this PIN code can be chosen by the customer.

A payment transaction flow using PayBox would be like this. (1) The customer gives her mobile phone ID to a merchant. (2) The merchant transmits the phone ID and the price of the purchase to PayBox by calling a special PayBox number or via the Internet. (3) PayBox calls the customer and a voice message asks for an authorization of the payment. (4) The customer authorizes the payment by entering her PayBox-PIN on her mobile phone. (5) After verification of the PIN, PayBox informs Deutsche Bank to settle the payment via the traditional payment system by placing a direct debit to customer's account. (6) PayBox credits the merchant. (7) The transaction is confirmed by an automated voice or SMS message sent to the merchant.

In PayBox, as in almost all existing mobile payment procedures (e.g., MobiPay, Caixamovil, Contopronto, Clear2-Pay, Easybuy, EMT, Mobiiliraha; see a survey in [18]), customer authentication is based on the mobile phone

[2] Although the described form of this service has been discontinued, it epitomises typical features of many, if not majority of, mobile payment systems and is by far the best documented. This is why we use PayBox as a basis for the analysis of security features of mobile payments.

ID. Indeed, in a mobile world, a mobile device is considered to be highly personal, belonging to and being managed by a single user; and the security of a mobile payment often relies on this perception. There is a reason for this: every GSM phone is equipped with a removable element called Subscriber Identity Module (SIM), which is a tamper-resistant microprocessor chip-card containing the identifiers, keys and algorithms required for a mobile network operator to authenticate a subscriber. Thus, in existing mobile payment procedures customer authentication rests, essentially, with network operators. End-user identification is based on the PIN code, and, in most cases, is done (if at all) once on power-on. Considering that most of the time the phones are "on", there is no reliable end-user authentication if phones are lost or stolen.

The transaction details are usually captured in an SMS (or even in voice) message which is sent in clear text. SMS messages can be easily forged by operator network insiders [20], and thus cannot ensure either confidentiality and data integrity nor non-repudiation.

In general, confidentiality and data integrity of payment transactions rely on transport-level security only. Independently of what interactive technology is used—voice, SMS, USSD, or WAP—there is no guarantee of end-to-end security as amply described in the Mobile Payment Forum paper [20].

UMTS (Universal Mobile Telecommunication System), a third generation mobile communication system that is being standardized by the 3G Partnership Project, brings several enhancements to the security features in GSM networks, such as data integrity and protection against the false base station attack [21]. However, as in GSM networks, unciphered connections are possible and UMTS does not offer any non-repudiation mechanisms for the mobile device. Use of PKI is specified with the use of IPsec between operators [22], but it does not extend to mobile devices. This means that additional measures have to be taken to ensure non-repudiation of exchanged messages even in UMTS networks.

To summarize, the shortcomings of most currently piloted mobile payment systems are:

- Authentication
  - Reliance on the personal nature of a mobile device.
  - Reliance on authentication by mobile network operators.
- Confidentiality and data integrity
  - Reliance on the communication technology/underlying mobile network security.
  - No true end-to-end security.
- Non-repudiation
  - Reliance on shared secrets possibly protected by PIN codes.

2.2.2. SAT-based security

A necessary condition for secure mobile payments is that the device in question has the capability of securely processing transactions over a wireless network. A SIM card can be seen as a trusted module within the mobile device. In an attempt to utilize SIM capabilities, the European Telecommunications Standards Institute developed a standard, called SIM Application Toolkit (SAT) [12], for value-added services and e-commerce in which GSM phones are used to initialize and authorize payment transactions.

Essentially, SAT enables a SIM card to drive a GSM handset interface, build up an interactive exchange between the network and the end-user, and access or control access to the network. Among mobile payment procedures security of which relies on SAT-enabled SIM cards are Mobilix, MobilBank, Oscar, and Fundamo (see survey in [18]). The customer must pre-register with the TTP/PSP and get a special PIN that "unlocks" the payment application on the SIM card. The mobile payment application on the card is initialized by programming into it a secret key shared between the TTP/PSP and the customer. The customer confirms payments by entering her PIN code, after which a transaction certificate is computed by encrypting the payment details using the shared secret key. Many SAT-based payment applications have the following characteristics:

(1) Based on symmetric cryptography (the secret key is shared between the customer and the TTP) for transaction certificates.
(2) Require cooperation with mobile network operators.
(3) More suitable for domestic markets due to limitations of secret keys.

The transaction certificate ensures confidentiality and data integrity but it does not guarantee non-repudiation. Because a secret key is shared between the customer and the TTP, it is possible for a customer to claim that the transaction has not taken place. Strong non-repudiation must employ digital certificates based on public-key cryptography. This is acknowledged by industrial consortia [11,1].

2.2.3. Bank card-based security solutions

Financial institutions prefer the security of mobile payments to be guaranteed by means of traditional bank cards for customer authentication and authorization of the transactions. The storage of the customer's credentials is of particular concern for viable mobile payment technology. Banks require that storage and processing of secret and private keys and other credentials must be based on tamper-proof hardware and that the execution environment (including OS and application software) has to be proved secure by a special process of Common Criteria certification (http://www.commoncriteria.org). This is why FSP-driven payment systems favour dual card or dual-slot phones or phones with an external card reader interface: the SIM card then can be used for customer authentication by a network, while the bank-issued smart card is used for storing customer credentials and for carrying on authentication procedures established by the bank.

EMPS (Electronic Mobile Payment Services) [23] is an example of such a system. It is a joint venture of Nordea bank, Nokia and Visa International. It requires a dual-slot WAP mobile phone with a WIM application. The second semi-permanent chip-card can be a debit or a credit card, and it is used for payment settlements. The payment can be made in virtual and real POS scenarios.

Usually bank authentication and authorization mechanisms are based on symmetric cryptography. This makes it easier for banks to personalize customers' bank cards and maintain the payment infrastructure. Authentication relies on two factors ("What you know" and "What you have") and dynamic challenge-response protocols; strong non-repudiation is rarely required due to the nature of the bank–customer relationship.

The disadvantage of card-based mobile payment procedures is that customers need to purchase a special (more expensive) phone and sometimes an additional device (card reader).

2.2.4. Secure mobile payments with PKI-enabled SIM cards

To understand the challenges of deploying WPKI in mobile payments, let us first consider three components that comprise public-key systems, namely public-key cryptography (PKC), personal security environment (PSE) and public-key infrastructure (PKI).

PKC is based on two keys, one for encryption and one for decryption. Only the decryption key must be kept secret: it is called the private key. The corresponding encryption key, called the public key, can be published. Computing private keys from their corresponding public keys is infeasible; this is the main property of PKC. The most widely used cryptographic algorithm is RSA [24].

A public-key management scheme works as follows. Each user is listed in a public directory, with her public key. If A wants to send a message to B, she obtains B's public key from the directory and uses it to encrypt the message. The encrypted message is sent to B. Only B is able to decrypt this message, because only B knows the corresponding decryption key. Furthermore, A can use her private key to digitally sign her message by encrypting the message (or its hash) with her private key. Such a message together with its digital signature can be sent over an insecure channel. Anybody can verify that (a) the message was indeed sent by A and (b) the message was not altered in a transmission, by looking up A's public key in the directory, encrypting the signature with this key and comparing the result with (the hash of) the original message.

Although everybody can read public-key directories, they must be protected from falsification and abuse. Therefore, an appropriate infrastructure, called a public-key infrastructure (PKI), must be set up for key distribution and management. The main components of a PKI are:

- The Certification Authority (CA) is responsible for issuing and revoking certificates for customers' public keys.
  - The Certificate contains a customer's public key and her personal details. It is digitally signed with the private key of the CA.
  - Customers of the PKI can validate digital signatures and their certificate paths from a known public key of a trusted CA.
- The Registration Authority (RA) provides a binding between public keys and the entities of their holders.
- Repositories store and make available certificate directories and a certificate revocation list (CRL).
- Directory service providers.

Private keys must be kept in a personal secure environment; usually, smart cards are used for this. The WAP standardization body (see http://www.wapforum.org/) devised a specification for secure provisioning and storage of the user's private and public-key pairs and certificates, namely Wireless Identity Module (WIM), which is generally accepted as an enabling technology for PKI in the context of mobile communication. It seems natural to consider a SIM card to be the most suitable candidate for WIM. Indeed, equipped with enough memory, 16- or 32-bit microprocessors, random number generators, and cryptographic accelerators, tamper-resistant SIM cards are capable of processing RSA efficiently. Technically, it is possible to implement strong customer authentication and non-repudiation using SIM cards, and technology-driven standardization bodies put WSIM forward as one of the possible approaches combining SIM and WIM applications on one chip-card [11]. Under the pressure of technology providers banks seem to soften their stance. In the latest white paper of the Mobey Forum it is explicitly stated that WIM can be "integrated in the mobile device, such as ... a SIM card-based solution" [1].

As was already pointed out, SIM cards have technical capabilities for securely storing and processing private keys. Common Criteria certification should not be a problem since the chip suppliers for smart cards and SIM cards are the same silicon vendors. However, the issue of initialization (i.e., equipping the card with the initial private–public key pairs and the root certificates) is not straightforward. While cards for dual-slot solutions could be delivered to customers pre-initialized by FSP/TTP, the situation is not so clear with SIM cards. These are owned by the mobile network operator, which hardly wants to concern itself with the cost of establishing and maintaining all or some components of the public-key infrastructure (i.e., providing services of certification and registration authorities), keeping and making available repositories of valid certificates and certificate revocation lists, etc. Moreover, even if one would be willing to do so, there will be an issue of interoperability.

One answer to this challenge of maintaining a PKI could be the use of a system supported by governmental bodies. In the following sections, we propose the use of a governmentally administered nationwide electronic identity PKI for mobile payments.

There are systems that utilize PKI, for example FairCash, SAP, and Safe Trader, but details concerning their security functions are scanty. Most of them are based on dual slot or dual card (with external card reader) solutions. We implemented our mobile payment system with the assumption of the PKI-enabled SIM card.

3. FINEID

In this section, we briefly describe the Finnish National PKI infrastructure, which is used in our mobile payment system for authenticating users and for ensuring the non-repudiation of payments. The Finnish electronic identity (FINEID) card along with the supporting infrastructure [25] is a system maintained by the Population Register Centre (PRC) of Finland. The card is a usual microprocessor smart card accepted as a valid ID in all EU countries and several others. The card contains two Citizen Certificates: the authentication certificate of the card holder and the signature certificate of the card holder. Private keys of both certificates are generated by the card and stored exclusively in its protected memory. Additionally, the card contains the PRC's own Certification Authority (CA) certificate.

The PRC maintains an online certificate directory (FINEID directory). Each registered individual gets a unique Finnish Electronic User ID (FINUID). The public keys of each user can be downloaded via a search with the appropriate criteria. Moreover, a revocation list of invalid certificates is available from the FINEID directory.

A mobile subscription customer can request a FINEID-compatible PKI-SIM card from their mobile network operator. At the moment, two operators in Finland issue such cards. The current implementation of FINEID on SIM cards is based on a platform developed by SmartTrust (http://www.smarttrust.com). The system works in the following way.

Card issuance. A customer receives a PKI-SIM card containing two private keys (generated on card) from her mobile network operator. The keys are stored within a SAT applet used for customer authentication. At this point, no public-key certificates corresponding to these keys exist. The certificates are produced and officially registered in the FINEID directory in the second step, when the customer registers her SIM card at a police station.

User authentication. In order to authenticate a customer, the service provider sends an authentication request to the mobile network operator's authentication service. The operator sends a challenge to the customer's phone in an SMS message. The headers of the SMS message instruct the phone to forward the message to the SAT applet, which produces a response to the challenge. Before signing the challenge, the SAT applet asks the user to enter her PIN code to access the private key. The response is sent back to the mobile network operator in an SMS message. The operator checks the response and informs the service provider of the results of the authentication.

The advantages of the SmartTrust platform include its compatibility with almost any GSM mobile phone, and the availability of all the infrastructure needed (the system is supported by three mobile network operators in Finland). However, there are also certain drawbacks in the current approach. For example, authentication cannot be done without the participation of the mobile network operator, and naturally, operators charge both customers and service providers for authentication. Apparently, this is a major threshold for joining the system: the system has been in place for almost two years, yet only a few service providers and fewer than 200 people use it.

We argue that mobile network operators can be eliminated from the authentication process. In the proposed mobile payment system the SmartTrust platform is *not* employed.

The user's private key residing on the FINEID card is used for signing product selections and payment orders, which are bound to each other. Signatures are calculated by the card, and a PIN code entry is used as a user authentication mechanism: prior to generating a signature, the user is asked to enter the PIN code corresponding to the signature certificate.

It is important to note that the signatures produced by the FINEID card are legally equivalent to the card holder's handwritten signature. Therefore, a signed combination of a product selection and a corresponding payment order constitute a legally valid agreement between the customer and the merchant.

FINEID certificates are only available to Finnish citizens and those permanently living in Finland. At the moment, a legal entity such as a merchant or a bank cannot obtain a FINEID certificate. In order for our platform to be fully open, certificates for such entities must be provided by a governmental structure (e.g., the National Trade Register), and the structure's root certificate (if it is different from the PRC's certificate) should be a part of the application. For simplicity, it is assumed in the description of our system that the CA for legal entities is the same as for individuals (i.e., the Population Register Centre).

4. Protocols

In this section, we describe protocols for two typical customer-to-business mobile payment scenarios: one for virtual, and one for physical point-of-sale payments. Other scenarios, for example, person-to-person payments, are not supported by the system. The notation used in the description of the protocols is given in Table 1.

Table 1
Protocol notation

CUST	Customer
MERCH	Merchant
BANK	Bank or another institution playing the roles of PSP and TTP
ID_X	The identity of subject X
SKEY_X	The secret RSA key of subject X
PKEY_X	The public RSA key related to SKEY_X
CERT_X	The public-key certificate of subject X
$\{m\}_K$	RSA encryption of the message m under the key K
MSG	A message
SIG	A digital signature
SIG_{X_Y}	A digital signature generated by X, intended to be verified by Y
$H(\cdot)$	A cryptographic hash function; we use SHA-1 in our protocols
PRODUCT	Product details
PRICE	Amount of money to be paid
ACCT_X	Account number of subject X
TIME	A timestamp
NONCE_X	A nonce generated by subject X

4.1. Virtual POS payment

A virtual POS payment is a remote payment (with no physical point-of-sale) which can be initiated by the user by browsing a merchant's website. Our protocol for a virtual POS payment contains the following steps (Fig. 2):

1. *Service request.* The customer initiates the protocol with the merchant by requesting product options. The request may contain information which limits possible options.

Customer → Merchant:

$$\mathrm{MSG} = \text{Service request}$$

2. *Service options.* The merchant sends a list of options to the mobile device. The list includes short descriptions of products and pricing information. The merchant also attaches its certificate to the list of options.

Merchant → Customer:

$$\mathrm{MSG} = \text{Service options}|\mathrm{CERT}_{\mathrm{MERCH}}$$

3. *Product selection.* The customer is prompted by the mobile device to select a product from the list. The information of the customer selection is sent to the merchant. The selection is signed using the customer's private key. The message is

Customer → Merchant:

$$\begin{aligned}\mathrm{MSG} &= \{\mathrm{ORDER}|\mathrm{SIG}_{\mathrm{CUST}_{\mathrm{BANK}}}\}_{\mathrm{PKEY}_{\mathrm{MERCH}}},\\ \mathrm{ORDER} &= \mathrm{PRODUCT}|\mathrm{NONCE}_{\mathrm{CUST}}|\mathrm{TIME}_{\mathrm{CUST}}|\\ &\quad \{H(\mathrm{PRODUCT}|\mathrm{TIME}_{\mathrm{CUST}})\}_{\mathrm{SKEY}_{\mathrm{CUST}}}|\\ &\quad \mathrm{ID}_{\mathrm{CUST}}|\mathrm{ID}_{\mathrm{BANK}},\\ \mathrm{SIG}_{\mathrm{CUST}_{\mathrm{BANK}}} &= \{H(\mathrm{TIME}_{\mathrm{CUST}}|\mathrm{ID}_{\mathrm{MERCH}}|\mathrm{PRICE})|\\ &\quad H(\mathrm{PRODUCT}|\mathrm{NONCE}_{\mathrm{CUST}})\}_{\mathrm{SKEY}_{\mathrm{CUST}}}\end{aligned}$$

where $\mathrm{NONCE}_{\mathrm{CUST}}$ and $\mathrm{TIME}_{\mathrm{CUST}}$ are a random nonce and a timestamp generated by the customer, PRICE is the amount of money the purchase will cost and PRODUCT is a string that describes product details.

4. *Payment request.* The merchant sends the payment details to the credit company. This payment information is signed using the merchant's private key and encrypted using the credit company's public key. The message in Phase 4 contains the merchant's details and payment details, such as amount, id of the customer and the signed message received in Phase 3.

Merchant → Bank:

$$\begin{aligned}\mathrm{MSG} &= \{\mathrm{REQUEST}|\mathrm{SIG}_{\mathrm{MERCH}_{\mathrm{BANK}}}|\\ &\quad \mathrm{SIG}_{\mathrm{CUST}_{\mathrm{BANK}}}\}_{\mathrm{PKEY}_{\mathrm{BANK}}}\\ \mathrm{REQUEST} &= \mathrm{ID}_{\mathrm{MERCH}}|\mathrm{ID}_{\mathrm{CUST}}|\\ &\quad \mathrm{TIME}_{\mathrm{CUST}}|\mathrm{PRICE}|\\ &\quad H(\mathrm{PRODUCT}|\mathrm{NONCE}_{\mathrm{CUST}})\\ \mathrm{SIG}_{\mathrm{MERCH}_{\mathrm{BANK}}} &= \{H(\mathrm{ID}_{\mathrm{MERCH}}|\mathrm{ID}_{\mathrm{CUST}}|\mathrm{TIME}_{\mathrm{CUST}}|\\ &\quad \mathrm{PRICE}|H(\mathrm{PRODUCT}|\\ &\quad \mathrm{NONCE}_{\mathrm{CUST}}))\}_{\mathrm{SKEY}_{\mathrm{MERCH}}}\end{aligned}$$

In this message $\mathrm{SIG}_{\mathrm{CUST}_{\mathrm{BANK}}}$ is the same signature as in Phase 3. After receiving this message the bank checks that the timestamp in the message is newer than the timestamp of the previous communication to detect any replay attacks.

It is possible for the merchant to sign a contract for processing mobile payments with a single acquiring bank. In this case the merchant sends the message to the acquirer, who has to pass the message to the bank, receive the payment confirmation (see the next phase) and forward it to the merchant.

5. *Payment confirmation.* The indicated amount of money is transferred from the buyer's account to the seller's account. Phase 5 is initiated by the credit company if

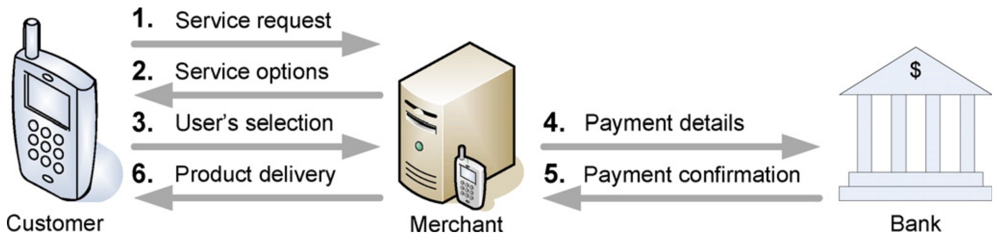


Fig. 2. Virtual POS payment model.

this transaction can be processed and finalized. The credit company sends a confirmation message to the merchant. The message is signed using the credit company's private key.

Bank → Merchant:

$$MSG = \{H(ID_{MERCH}|ID_{CUST}|\ TIME_{CUST}|PRICE|\ H(PRODUCT|NONCE_{CUST}))\}_{SKEY_{BANK}}$$

From this message the merchant can check that the payment was made with the agreed amount from the account of the customer to the account of the merchant. The hash value $H(PRODUCT|NONCE_{CUST})$ is meant for the customer to make sure that the merchant can not claim that the customer bought something other than the intended product.

6. *Product delivery.* Finally, the merchant checks the message received in Phase 5. If the message is valid and the payment has been made, the merchant delivers the product to the customer. The merchant also sends the customer a message stating that the payment has been made and the product has been sent.

Merchant → Customer:

$$MSG = \{H(ID_{MERCH}|ID_{CUST}|\ TIME_{CUST}|PRICE|\ H(PRODUCT|NONCE_{CUST}))\}_{SKEY_{BANK}}$$

The customer can check that the amount of money, product details, the nonce and the timestamp all match the original values to be sure that the correct amount was paid for the correct product.

4.2. Real POS payment

Now we consider a proximity payment which involves a real point-of-sale terminal, e.g., a vending machine. Our protocol for a secure real POS payment assumes an underlying wireless technology such as Bluetooth [26], and contains the following steps (see Fig. 3).

1. *Initiation.* The customer initiates the protocol with the merchant by choosing a product. If the vending machine supports several ways of payment, the user may need to explicitly select the mobile payment option. Optionally, the protocol can be initiated by the vending machine, which detects the device when it comes within the range of the Bluetooth communication. No messages are sent in this phase.
2. *Bluetooth pairing.* To enable the exchange of messages, Bluetooth pairing must be performed between the vending machine and the mobile phone. If several Bluetooth devices are in the range, the machine can use a random PIN code for pairing and show this PIN on its display. The user must enter this PIN code in the mobile phone.
3. *Product offer.* If the user has not selected a product yet, the vending machine sends a message with information about available products and their prices. If Phase 1 was initiated by the user, and the product is already selected, the list of products contains only the selected item. In addition to this data, the vending machine sends its own certificate $CERT_{MERCH}$ and a random nonce $NONCE_{MERCH}$.

Merchant → Customer:

$$MSG = CERT_{MERCH}|NONCE_{MERCH}|\text{List of products}$$

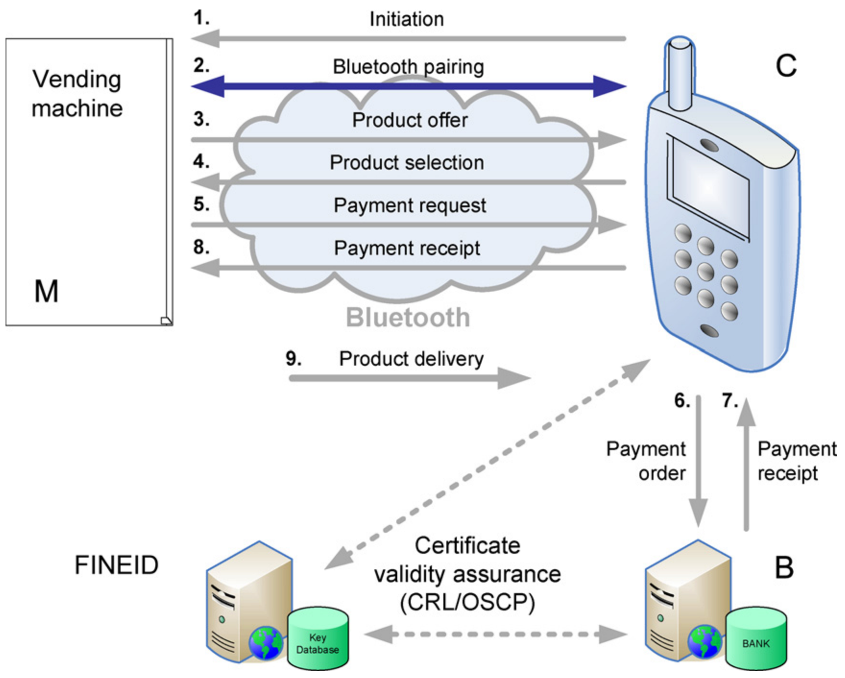


Fig. 3. A mobile payment protocol for vending machines.

After receiving the message, the customer extracts the merchant's certificate and checks its validity.

4. *Product selection.* The user is prompted by the mobile device to select a product, unless it has already been selected. The information of the user selection is sent to the vending machine. Also, the customer's certificate is included in the message. The mobile phone must store the price PRICE of the selected product, as it will be needed later for payment.The message in Phase 4 consists of three parts. The first part is the user's selection PRODUCT, and a nonce $\text{NONCE}_{\text{CUST}}$ generated by the mobile device. This part of the message is encrypted with the vending machine's public-key $\text{PKEY}_{\text{MERCH}}$. Second, the user's certificate is appended to this message. The last part of the message is a signature SIG which binds together the product selection and the two nonces $\text{NONCE}_{\text{CUST}}$ and $\text{NONCE}_{\text{MERCH}}$.

Customer → Merchant:

$$\begin{aligned}\text{MSG} &= \{\text{PRODUCT}|\text{NONCE}_{\text{CUST}}\}_{\text{PKEY}_{\text{MERCH}}}|\\ &\quad \text{CERT}_{\text{CUST}}|\text{SIG}\\ \text{SIG} &= \{H(\text{PRODUCT}|\text{NONCE}_{\text{MERCH}}|\\ &\quad \text{NONCE}_{\text{CUST}})\}_{\text{SKEY}_{\text{CUST}}}\end{aligned}$$

After receiving the message, the merchant extracts the customer's certificate and verifies it. Then the merchant decrypts the message, obtaining information about the user's selection PRODUCT and $\text{NONCE}_{\text{CUST}}$. Finally, the merchant verifies the signature using the customer's public key. The vending machine could also check the certificate revocation list to see that the user certificate has not been revoked, but this checking can also be made the responsibility of the bank.

5. *Payment request.* The vending machine sends a payment request to the mobile device. The request is signed using the vending machine's private key $\text{SKEY}_{\text{MERCH}}$.The payment details include the account number $\text{ACCT}_{\text{MERCH}}$, and a reference id of the vending machine ID_{MERCH}. Note that the price of the product is not sent with the payment details, since the customer already knows it. However, it is included in a hash in the second part of the message. The customer's certificate, the price of the product, and two nonces $\text{NONCE}_{\text{MERCH}}$ and $\text{NONCE}_{\text{CUST}}$ are concatenated, hashed, signed with the vending machine's private key and appended to the message MSG. The last part of the message is the merchant's signature.

Merchant → Customer:

$$\begin{aligned}\text{MSG} &= \text{REQUEST}|\text{SIG}\\ \text{REQUEST} &= \text{ACCT}_{\text{MERCH}}|\text{ID}_{\text{MERCH}}|\\ &\quad \{H(\text{CERT}_{\text{CUST}}|\text{PRICE}|\text{NONCE}_{\text{MERCH}}|\\ &\quad \text{NONCE}_{\text{CUST}})\}_{\text{SKEY}_{\text{MERCH}}}\\ \text{SIG} &= \{H(\text{REQUEST})\}_{\text{SKEY}_{\text{MERCH}}}\end{aligned}$$

The customer will later send the signed hash of her certificate, the price and both nonces to the bank. The bank will use (and optionally store) this as proof of transaction authorization by vending machine. This way the customer cannot offer one certificate to the merchant and another to the bank, or change the amount to be paid. The signed hash can also be stored by the customer as a receipt from the vending machine. Combined with a receipt from the bank (see Phase 7), this can be used later as proof of purchase if a dispute arises.

After receiving the message MSG, the customer verifies the signature SIG using the merchant's public key.

6. *Creation of a payment order.* The customer sends a payment order to the bank. In addition to the information received from the merchant in the previous steps, an account number of the customer $\text{ACCT}_{\text{CUST}}$ and a timestamp TIME are needed for the transaction.
   The customer creates a payment order and sends it to the bank encrypted with the bank's public key and signed with the customer's private key.

Customer → Bank:

$$\begin{aligned}\text{MSG} &= \{\text{PO}\}_{\text{PKEY}_{\text{BANK}}}|\text{SIG}\\ \text{PO} &= \text{ACCT}_{\text{CUST}}|\text{ACCT}_{\text{MERCH}}|\text{ID}_{\text{CUST}}|\text{ID}_{\text{MERCH}}|\\ &\quad \text{PRICE}|\text{NONCE}_{\text{MERCH}}|\text{NONCE}_{\text{CUST}}|\text{TIME}|\\ &\quad \{H(\text{CERT}_{\text{CUST}}|\text{PRICE}|\text{NONCE}_{\text{MERCH}}|\\ &\quad \text{NONCE}_{\text{CUST}})\}_{\text{SKEY}_{\text{MERCH}}}\\ \text{SIG} &= \{H(PO)\}_{\text{SKEY}_{\text{CUST}}}\end{aligned}$$

In this message everything except the customer's account number was received from the merchant in the previous stage. The bank is obviously the one where the mobile phone user has an account.

Here we assume that the mobile device already has the bank's public key. Certificates of participating banks can be installed into the device when the software is installed. We can also include a procedure for importing a certificate of a bank which has joined the protocol after the software was installed.

7. *Payment processing.* After receiving and decrypting the payment order, the bank verifies the customer's signature which is attached to it. For this, the bank retrieves the customer's certificate from the FINEID directory; ID_{CUST} is used as the search key. In the same way the bank gets the merchant's certificate in order to verify the merchant's signature on the payment order. This is done to make sure that the same customer's certificate and nonces were used in communication between the customer and the merchant. In addition, the bank checks that neither certificate is on the revocation list. The bank also compares the timestamp to the stored timestamp of the previous payment order received from the customer (if any) to defeat replay attacks. Upon successful passing of all checks, the bank transfers the

amount of money from the customer's account to the merchant's account. If the merchant's account is in another bank, the usual interbank procedures are used for crediting money to the merchant. If the transaction can be processed and finalized, the bank sends a confirmation message (receipt) to the mobile phone.

The receipt provides proof that the payment has been made. The bank account number of the vending machine, the amount of money and nonces $\mathrm{NONCE_{MERCH}}$ and $\mathrm{NONCE_{CUST}}$ are hashed and signed using the bank's private key:

Bank → Customer:

$$\mathrm{MSG} = \{H(\mathrm{ACCT_{MERCH}}|\mathrm{PRICE}|\mathrm{NONCE_{MERCH}}|\mathrm{NONCE_{CUST}})\}_{\mathrm{SKEY_{BANK}}}$$

The customer has all the information needed for calculating of the same hash and verification of the bank's signature.

8. *Proof of payment.* In Phase 8, the mobile phone forwards the bank receipt to the vending machine. In order to specify which bank's public key must be used for verification of the receipt, the bank's id $\mathrm{ID_{BANK}}$ is included in the message.

Customer → Merchant:

$$\mathrm{MSG} = \mathrm{ID_{BANK}}|\{H(\mathrm{ACCT_{MERCH}}|\mathrm{PRICE}|\mathrm{NONCE_{MERCH}}|\mathrm{NONCE_{CUST}})\}_{\mathrm{SKEY_{BANK}}}$$

We assume that the vending machine already has certificates of participating banks. Therefore, the vending machine can decrypt the receipt using the bank's public key. The vending machine then calculates hash

$$H(\mathrm{ACCT_{MERCH}}|\mathrm{PRICE}|\mathrm{NONCE_{MERCH}}|\mathrm{NONCE_{CUST}})$$

and verifies that its value is the same as in the receipt.

The vending machine must have a list of valid public keys of different banks. If the vending machine does not have a network connection, updating and revoking bank certificates may be cumbersome. The protocol may be extended to check the validity of bank certificates by forwarding Online Certificate Status Protocol (OCSP) requests through the mobile phone to a trusted server.

5. Testbed environment, implementation and architecture

As a proof-of-concept, we designed and implemented a system for buying train tickets using a mobile phone. The client software in the buyer's phone was implemented using Java 2 Micro Edition (J2ME). The main advantages of J2ME over competing technologies are better portability and somewhat easier implementation. J2ME applications, called MIDlets, are run in a Java virtual machine. This facilitates the use of the Java sandbox security paradigm, enhancing the user perception of the security of the program. The client application was implemented using the Sun WTK (Wireless Toolkit) version 2.2 from Sun Microsystems.

5.1. Client functionality

The functionality of the client software is depicted in Fig. 4. The execution flow is divided into phases according to Section 4.1. Phases that occur in the mobile device are circled. When the application is started, two certificates, one of the CA and another one of the OCSP server, are read from the local storage. This storage is the resource folder of the application's Java archive (JAR) file. RSA public keys are read from both certificates for later use.

Phase 1 of the program execution displays on the screen of the mobile phone a form where details of the journey (e.g., departure and destination, date and desired time) can be entered. While the user is filling the form fields, a separate thread sends an OCSP request querying the status of the merchant certificate. Finally, the journey details entered by the user are submitted to the merchant.

In Phase 2, the travel options received from the merchant are shown on the display as a list, and the user is prompted to select a journey. As the form with the travel options is shown, two threads are started: one for retrieving the bank certificate and a subsequent thread for retrieving the OCSP status for that certificate. It should be noted that the bank certificate can be stored in the phone and retrieving it is not always necessary.

In Phase 3, the payment order message is created (details of the message format are given in Section 4.1). The user is prompted to enter the PIN code related to the signature certificate on her FINEID card, to sign the payment order. The merchant forwards the payment order to the bank in Phase 4 and receives a receipt in Phase 5. The merchant then forwards the receipt to the customer.

In Phase 6, the application decrypts the payment receipt generated by the bank. The bank's signature on the receipt is verified. The client also checks whether the receipt corresponds to the payment order sent in Phase 3 and notifies the user of the result of the transaction. See Fig. 5 for an example of the payment transaction.

In our implementation we made the decision that only a single merchant is supported. However, the application can be made more general by allowing the user to select a merchant in Phase 0. Available merchants can be queried from a central portal, and a response message would include instructions on what kind of user interface is to be generated for buying from a given merchant. The user interface would then be dynamically created according to these instructions, and subsequent messages would be directed straight to the right merchant.

Similarly, if the user has more than one bank available for transaction settlement, she could be prompted to select the bank in Phase 3. After the user has selected the desired

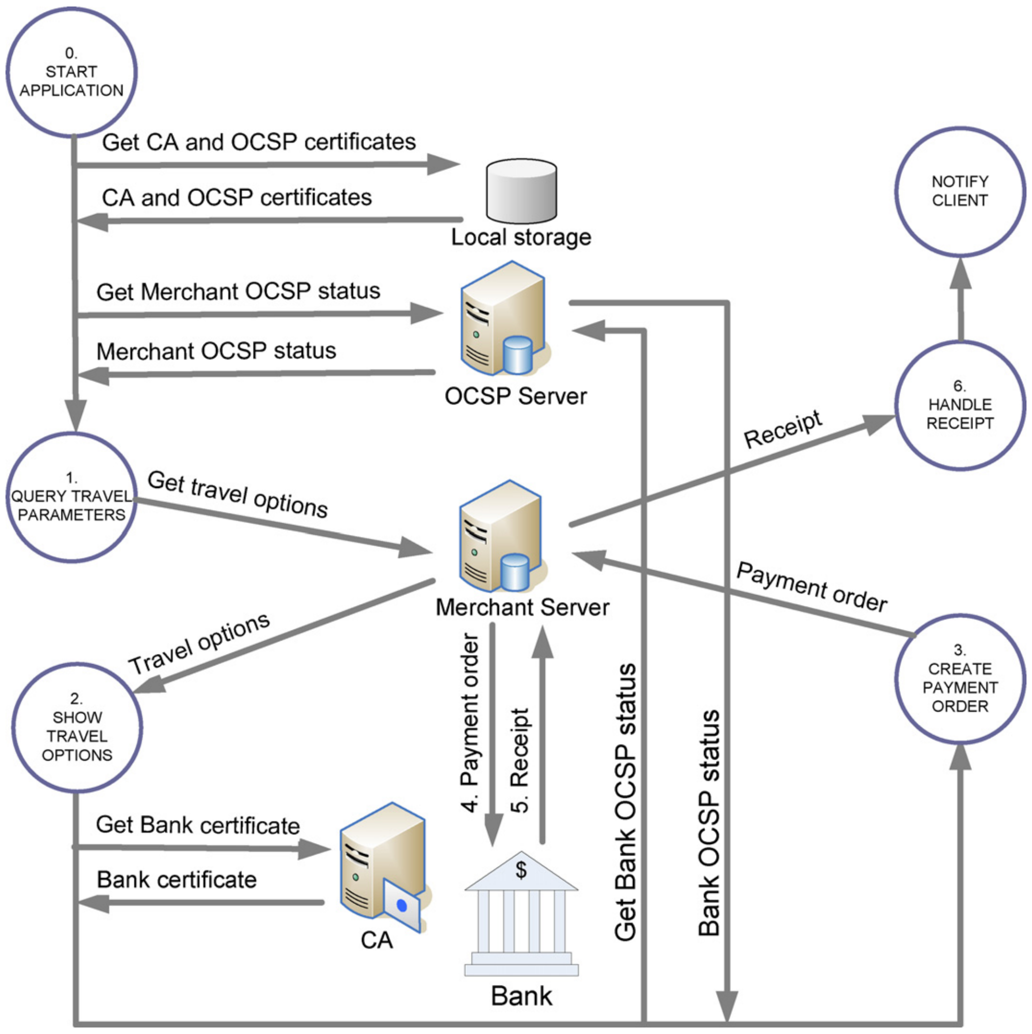


Fig. 4. Application for buying train tickets: overall scheme of the implementation.

bank, the bank certificate and subsequent OCSP status are retrieved before entering the next phase.

5.2. Communication

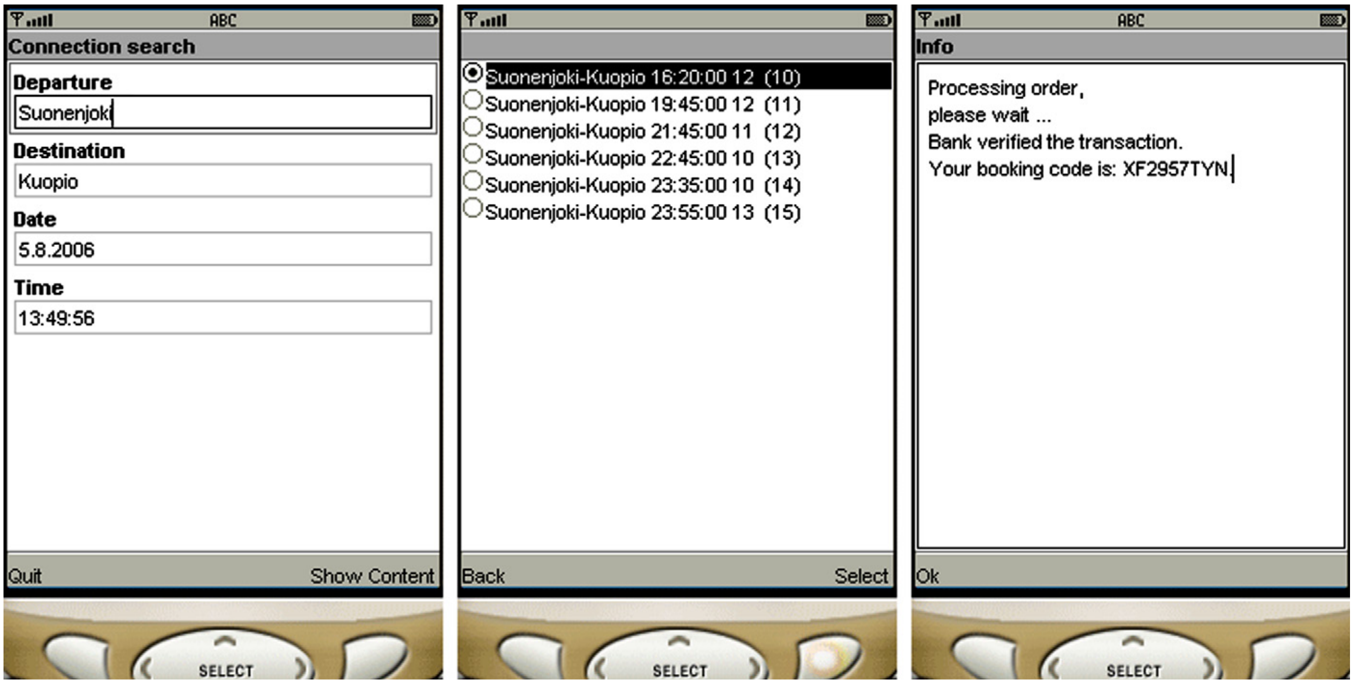
Implementing the communication between the mobile user and the merchant was done using the Web Services technology. Message exchange was implemented using SOAP (Simple Object Access Protocol), which is an XML-based technology for message exchange over HTTP. In J2ME there is an optional SOAP API (JSR-172) [14] that has been implemented in a number of devices available on the market today, such as the Nokia N91, which we used for testing our implementation.

For communicating with a Web Service, stub classes are included in the MIDlet. Stubs are responsible for creating outgoing messages and parsing incoming messages. Parameter marshalling is done by stubs in such a way that all the programmer needs to do is to call an appropriate method with suitable parameters. Moreover, stubs can also handle parameter arrays or even more complex parameter types, which are represented as classes in J2ME. The WTK contains a stub creator utility that takes a WSDL (Web Service Definition Language) file or an URL as a parameter and creates the necessary stub classes. In our prototype implementation, a singe train connection is a complex type represented as a Connection object in J2ME, and the list of train connections is passed as an array of Connection objects.

5.3. Bouncy castle

The J2ME environment does not contain classes for cryptographic processing, such as encryption or handling public-key certificates. However, there is a package available from the Legion of Bouncy Castle (http://www.bouncycastle.org) that can be used in the J2ME environment. The package is commonly referred to as the Bouncy Castle Lightweight API.

The Bouncy Castle package provides classes for encryption and decryption, calculating digests, and handling certificates, among others. For example, for digitally signing messages in various phases of the protocols, we used the classes SHA1Digest and RSAEngine available in the



Connection search

Departure
Suonenjoki

Destination
Kuopio

Date
5.8.2006

Time
13:49:56

```
(●) Suonenjoki-Kuopio 16:20:00 12  (10)
( ) Suonenjoki-Kuopio 19:45:00 12  (11)
( ) Suonenjoki-Kuopio 21:45:00 11  (12)
( ) Suonenjoki-Kuopio 22:45:00 10  (13)
( ) Suonenjoki-Kuopio 23:35:00 10  (14)
( ) Suonenjoki-Kuopio 23:55:00 13  (15)
```

Info

Processing order,
please wait ...
Bank verified the transaction.
Your booking code is: XF2957TYN

Fig. 5. Main displays of the application.

API. However, there are no ready-made tools for certificate signature verification in Bouncy Castle. In our application, a certificate is verified by calculating a hash of the TBS (to be signed) structure of the certificate and comparing the result with the decrypted signature of the certificate.

Open certificate status protocol (OCSP) [13] was used for obtaining certificate revocation status messages. The Bouncy Castle has a number of ready-made classes for creating OCSP requests. In the OCSP protocol, the request contains a TBSRequest structure and an optional signature, which was not used in our case. We used the OCSP-Request class to generate requests. A TBSRequest contains a version identifier, an optional name of the requester, a list of requests and an optional list of extensions. The requests in the list were created using an array of Bouncy Castle Request objects and the extension using the X509Extensions class. OCSP requests were sent using the HTTP protocol and were written into the output stream with the ASN1OutputStream class.

5.4. The application package and deployment

The lightweight package of the Bouncy Castle has some classes which belong to system packages, for example, the BigInteger class in package java.math. The class loader does not allow such classes to be loaded, as they could be used to replace system package classes and thus would get more privileges than desired [27].

Obfuscation is a method that renames classes to make the re-engineering of source code from class files more difficult [28]. As classes that would otherwise conflict with system package names are also renamed, they can be used without the danger of being classified as system package classes. Obfuscation also removes unnecessary classes, which often makes the resulting JAR packages smaller. As the Bouncy Castle lightweight API has a lot of classes, the JAR package for our project is 587.4 K before obfuscation and 81.9 K after obfuscation. We used the proguard obfuscator for obfuscation.

Deploying a J2ME MIDlet into a device can be done in several ways. The application can be uploaded into the device using a cable, Bluetooth or IrDA, or it can be downloaded from the Internet using OTA (Over The Air) provisioning. For the end-user the last option is by far the easiest to use. For our device testing we set up an OTA download page at http://katiska.uku.fi/~mhassine/OTA/MobileVR.html.

5.5. Performance and bottlenecks

Computation speed was tested on a Nokia N91. Retrieving connections that matched the search criteria from the server using GPRS communication took an average of 3.2 s. An average of 8.6 s was required to create and send the order message and to receive and verify the response message. Some of the response time was consumed by retrieving and verifying certificate information. After this communication was moved to an earlier stage, the response time decreased to an average of 7.5 s.

To evaluate the performance in the context of current payment systems, we observed the procedure of paying in a shop with a credit/debit card. We measured the time needed for a customer to hand out the credit card, the clerk to read the card and print out the receipt, and the customer to sign the receipt. The shortest times recorded were a little over 15 s with average being over 20 s. All of these purchases were of small monetary value, less than 50€. With purchases worth more than 50€, the time needed to finalize a purchase is much longer as the terminal dials the bank to inquire whether the user has a sufficient balance to pay for the purchase. Paying with a credit card in the Internet is even slower as it typically requires the user to type in the credit card number.

As the number of users of a mobile payment system increases, the number of payment requests becomes too large for a single server to handle. Fortunately, the number of payment servers can be increased with relative ease. As the payment system described in this paper relies on a single CA, the performance of the CA server can be restrictive. The CA is in general responsible for both issuing new certificates and maintaining a CRL. The CRL is needed by several parties in the protocol and hence would be downloaded very often, unless OCSP is used. As OCSP servers usually base their function on CRL lists, a single OCSP server can serve a large number of clients and needs to retrieve the CRL only whenever a new list is out. A number of OCSP servers can be set up to serve the clients of a single CA. This makes the OCSP approach very scalable in comparison with using a CA or an RA to distribute a CRL.

5.6. Smart card applet

For connection to a SIM card with FINEID functionality we used the Security and Trust Services API (SATSA, JSR-177) [29] in our architecture. This limits the functioning of the system to relatively new mobile phone models which implement SATSA. However, this ensures that the system is not bound to a given mobile network operator infrastructure. We used the usual bankcard-sized FINEID card in our tests, because PKI-SIM cards currently deployed by Finnish mobile network operators do not allow straight communication between the phone software and the SIM card. This must be changed in order for our system to be fully functional. In what follows we describe an architecture for the FINEID-SIM suitable for our mobile payment system.

FINEID must be implemented on the SIM card as an (U)SAT applet. It must be noted that the applet is installed on the card by the mobile network operator or by the supplier of SIM cards. After installation, the applet is initialized. It generates two RSA keypairs and sends the public keys to the terminal. Certificates containing these public keys are generated, signed by the CA and transferred to the applet along with the root CA certificate. The private keys are stored exclusively by the applet. The user cannot install new applications on the card or remove existing ones. Therefore, the SIM card remains a trusted token.

The functionality of the applet includes the computation of digital signatures, deciphering, and hashing. Access to the private keys is provided upon correct entry of the corresponding PIN codes. The user's public-key certificates can be retrieved from the applet.

In the (U)SIM card there must be an access control entry with a hash of the operator domain certificate. The implementation of SATSA on the mobile phone compares this hash with that of the certificate used for signing the J2ME applet. The J2ME applet is allowed to exchange messages with the FINEID applet only if the hashes match. This security mechanism is in place to ensure that the J2ME applet has not been tampered with.

The J2ME applet sends requests to the FINEID applet using ENVELOPE APDUs (see [12] for the details on the messages format). When a response must be retrieved, the GET RESPONSE command is used. The functions of the FINEID applet are encoded in the INS bytes of APDUs.

The environment that we used for testing our application is depicted in Fig. 6. The J2ME applet running in Sun WTK is implemented to communicate with the FINEID card using SATSA. However, this communication is forwarded by WTK to a bankcard-sized FINEID placed in a smartcard reader. We implemented a small mediator program which accepts commands arriving from WTK (they are sent as messages of TLP224 protocol), sends them to the card, and forwards the card's responses back to WTK.

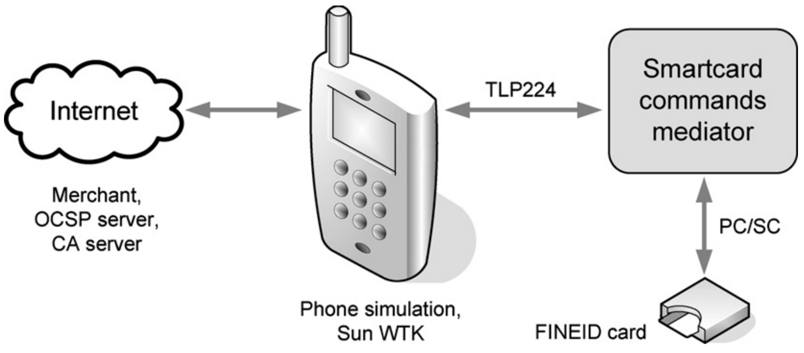

Internet
Merchant, OCSP server, CA server
TLP224
Phone simulation, Sun WTK
Smartcard commands mediator
PC/SC
FINEID card

Fig. 6. Testbed environment.

5.7. Simulation software for the merchant and the bank

The merchant server was implemented using the PHP script language. The NuSOAP (http://dietrich.ganx4.com/nusoap/) package was used to create a single PHP script that has methods for handling the request messages sent by the J2ME client. The SOAP server was created as an object, and the WSDL (Web Service Definition Language) (http://www.w3.org/TR/wsdl/) definition was automatically created by the NuSOAP package after the methods and complex data types had been registered into the server object.

Because FINEID certificates are not available for legal entities, we created our own CA to produce certificates for the merchant, the bank and the OCSP server in our simulation. The CA and OCSP certificates were included in the client JAR package so that the client could verify the certificates of other parties. The CA and all certificates were created using OpenSSL (http://www.openssl.org/), and keys were created using JDK's keytool. A short Java program was used to extract the private keys for the bank and the merchant from the keystore. The certificates were converted to Privacy Enhanced Mail (PEM) [30] format using OpenSSL, because the PHP OpenSSL module expects to have certificates in PEM format.

Cryptographic processing in the PHP scripts was done using the PHP OpenSSL module. In encryption and decryption PKCS1 padding was used since the FINEID card uses this padding in digital signatures [31].

The OCSP server was implemented as a Java Servlet. This decision was motivated by the easier implementation of OCSP functionality utilizing the ready-made OCSP classes of Bouncy Castle. With servlets it is possible to use the broader API including the Java cryptography extensions (JCE) classes, which are not available in the Lightweight API for J2ME. A ready-made OCSP server package is also available from Novosec (http://sourceforge.net/projects/novosec-bc-ext/).

6. Conclusions and future work

Many current mobile payment systems rely on mobile network operators for authentication, and lack adequate non-repudiation. As stated in the introduction, for strong authentication and non-repudiation the use of public-key cryptosystems is required. In this work, we investigated whether a national PKI infrastructure can be used for facilitating mobile payments. Section 6.1 summarizes the contributions of this paper and Section 6.2 discusses the measures needed to turn this system into a commercial system. Future work is described in Section 6.3, and in Section 6.4 prospects developing a pan-European system based on national PKI infrastructures are discussed.

6.1. Contributions

In this work we have designed a mobile payment system that uses a governmentally administered public-key infrastructure, namely, the Finnish electronic identity. Using FINEID, the system authenticates persons, not customers of a certain bank, mobile network operator, or payment service provider. Moreover, it ensures non-repudiation of payments and also provides integrity and confidentiality of the messages related to the payment transactions. Furthermore, as the administration of the PKI system is the responsibility of the government, the system is very economical for both the service providers and the users.

We described two mobile payment protocols, one for virtual point-of-sale payments, and one for physical point-of-sale (vending machine) payments. The proof-of-concept implementation, a system for buying train tickets, was done using freely available development tools and platforms. We have used J2ME for developing mobile phone software, and PHP for developing server-side software. One of the main motivations for choosing J2ME was its portability across different device platforms. Implementing payment systems on the J2ME platform has been shown to be feasible; the computation speed is satisfactory, and the application is of reasonable size. The delays experienced by the customer when using the mobile payment application are acceptable.

6.2. Turning the system into a commercial service

In order to launch the system described in this paper, a number of changes to the FINEID infrastructure are needed. The *Population Register Centre (PRC)* currently issues certificates only to citizens. In order to have a certificate on a SIM card, a *customer* must get a SIM card from her mobile network operator and register it at a police station. The *Mobile network operators* have to change the FINEID applet on SIM cards in such a way as to enable access to it from the authorized phone software. The requirements for such SAT-based FINEID applet are described in Section 5.6. The customer must also download and install a J2ME program on her mobile phone in order to be able to use the proposed mobile payment system. To make our system feasible, the PRC has to start issuing certificates to business entities, such as merchants and banks. In addition, it has to set up an OSCP server for checking the validity of certificates.

In order to support virtual POS payments, the *merchant* has to implement a web service ready for processing mobile payments. Technologies suitable for building such service are described in Section 5.2. To support real POS payments using a mobile phone, a vending machine has to implement the protocol described in Section 4.2. To enable the protocol between the customer and the *bank*, the bank must implement the protocol and accept authentication with the FINEID card. At least one bank group in Finland (Osuuspankki) already supports authentication of its customers using FINEID cards. In order to support mobile payments with PKI-SIM cards, additional software must be implemented and deployed at the bank servers. It is not necessary to connect the vending machine to the Internet or other network.

6.3. Future work

When the Security and Trust Services API is used for accessing a SIM card with sensitive data such as user's private keys, attention must be paid not only to the security of the SIM card, but also to the security of the J2ME program accessing the card. The integrity of the program is protected (see Section 5.6 for details). However, if a vulnerability is found in such a program, it could be used for signing illicit payment orders. Therefore, mechanisms for blocking vulnerable versions of programs and for updating them with newer versions must be devised. This issue is an active research area with technology providers. For example, the Open Mobile Alliance (http://openmobilealliance.com) has a special working group dedicated to deriving specifications for secure over-the-air updates.

Another issue that has to be investigated is the standardization of the mobile phone as a payment card terminal. Because a SIM card is used for facilitating payments in our system, Europay-Mastercard-Visa (EMV) or similar standards might require the standardization of mobile phone keypads for entering payment-related PIN codes. As was already mentioned, at least one Finnish bank group accepts FINEID cards for the authentication of its customers. Among other uses, such authentication is accepted for confirming 3D Secure credit card payments. In other countries more strict standards may be in force.

6.4. Implementation on the level of the european union

Many mobile payment systems can be extended to work seamlessly within several countries. To achieve this, the necessary infrastructure must be introduced, and contracts between mobile network operators may have to be established. Using governmentally supported PKI in a payment system is restrictive in a world where people travel a lot. Several European countries are implementing their eID (electronic identity) systems using PKI. On the European level, the European Commission has launched an interoperability action plan to obtain interoperable electronic identity management by 2010 [32]. When implemented, the plan will enable each member state to govern its own PKI-based eID system, while providing interoperability within the EU. Such an interoperable electronic identity can help create a pan-European mobile payment system based on electronic identity.

The Liberty Alliance Project (http://projectliberty.org) is a consortium in which governmental bodies, academia and businesses cooperate in the research and standardization work on federated identity management. The alliance has developed a number of open technical, business and privacy standards for identity-based services. Among them are standards and guidelines for mobile industry [33]. Participation of the European government bodies in the Liberty Alliance Project provides a promising outlook for the future of utilizing national public-key infrastructures in identity-based services, including mobile payment systems. National electronic identity cards are already being used in payment services, for example, Internet banking applications. However, such cards are not readily applicable for mobile payments using one-slot mobile phones, which are the most common.

Clearly, building a national or a pan-European PKI-based mobile payment system requires support from the authorities: namely, it should be possible for the user to acquire a legally valid certificate on a SIM card. Currently, only in Finland and in Germany can one get such a legally valid certificate from a mobile network operator. In Finland, the certificate is signed by the Population Register Centre, while in Germany the mobile network operator acts as a certification authority if it has received an official license to do so. Both scenarios are acceptable in our MP system, provided that the certificate directories are open.

An IST project SEMOPS [34], funded by the European Union, aims at creating a pan-European payment system. The project is based on the collaboration of banks and mobile network operators, with the objective to build a payment system that can deliver a variety of payment services to customers of participating banks and operators. The project is launching a pilot in Greece, Italy and Hungary to deliver interbank and cross-border payment services (http://www.semops.com/).

The scope of our work is, however, somewhat different. SEMOPS aims at delivering an open and modular payment system that supports various security modules (including ones based on public-key infrastructures). Our approach, being complementary to SEMOPS, aims at providing mobile payment services as a natural part of utilizing a pan-European or even a worldwide identification infrastructure. Governments have been granting means of identification for their citizens for centuries in forms of passports and other identification documents. An electronic identity is a natural continuation of such means of identification, and we believe that for many applications it will eventually replace existing paper documents.

As banks start utilizing electronic identity as a means of authentication, the use of electronic IDs in payments will pave the way for mobile payments based on a European or even a worldwide electronic identity.

Acknowledgements

The authors thank the anonymous reviewers for their constructive criticism on earlier drafts of this paper. We thank Adi Shamir for a discussion and helpful comments on our work.

Appendix A. Table of mobile payment systems

Security basis	Authentication	Confidentiality and data integrity	Non-repudiation	Examples of MP systems
Phone-based	Phone ID + Service PIN	Security is based on underlying communication technology • No end-to-end security • Data sent via SMS or voice call-back in clear	Static PIN	Brokat Twister X-Pay, EasyBuy, EMT, Metax, Mint, Mobiiliraha, MobilMat, MobilPay, Pay@Once, PayBox, Paydirect-Yahoo, PayPal, Quick@More, Simpay, Skidata, SmartPAY Echovox, StreetCash, Telemoney, Trivnet, VisaMovil
			TAN sent by SMS or voice	m-till, MobiPay, O-card, PaybyTel, Sonera Shopper, Turkcell
Software on phone • Client wallet	Phone ID + password-protected memory	Data sent by • SMS • Form filling	PIN or TAN	Pay-by Phone/T-pay i-mode, Nokia m-wallet, PaymentWorks, SmartMoney, ZOOP (+IrFM)
• Server wallet	+ login + password			Contopronto, Genion m-payment, Nokia m-wallet
SAT • multi-application	Phone ID + application-specific PIN	Data encrypted with a shared secret key	Transaction certificate encrypted with a shared secret key	BeamTrust (+IrFM), Fundamo, KDDI (+IrFM), m-Maestro, Mobilix, Oscar, Sonofon m-Banking
• W/SIM (dual chip)				Solo
Dual card • dual slot	Phone ID + two factors: • what you have (card) • what you know (PIN)	Data encrypted with a shared secret key	Transaction certificate encrypted with a shared secret key	CyberCOM, DoCommerce(FeliCa), EMPS, Iti Achat, Nokia/Master Card
• external card reader				Mobile, Paiement CB Sur
PKI • W/SIM dual chip	Phone ID + service PIN + digital certificate	Data encrypted with a session key	Digital signature with a private key	FairCash (dual chip)
• SIM-WIM dual slot, dual card or external card reader				Macalla, MobilHandel SmartPay, SPA
• Server-based				Nokia Payment, SafeTrader, Solution

Appendix B. List of acronyms

Acronym	Meaning
3G	3rd generation (in the context of mobile phone standards)
API	Application Programming Interface
CA	Certificate Authority
CRL	Certificate Revocation List
EMPS	Electronic Mobile Payment Services
EMV	Europay-Mastercard-Visa
EU	European Union
FINEID	Finnish Electronic Identity
FINUID	Finnish Electronic User ID
FSP	Financial Service Provider
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HTTP	Hypertext Transfer Protocol
IrDA	Infrared Data Association
IST	Information Society Technologies
J2ME	Java 2 Mobile Edition
JAR	Java Archive
JDK	Java Development Kit
JCE	Java Cryptography Extension
OCSP	Open Certificate Status Protocol
OTA	Over The Air
PDA	Personal Data Assistant
PEM	Privacy Enhanced Mail
PHP	Personal Home Pages (a scripting language for dynamic web page creation)
PIN	Personal Identification Number
PKC	Public-Key Cryptography
PKCS1	Public-Key Cryptography Standard #1
PKI	Public-Key Infrastructure
POS	Point-of-Sale
PRC	Population Register Centre
PSE	Personal Security Environment
PSP	Payment Service Provider
RA	Registration Authority
RFID	Radio Frequency Identification
RSA	Rivest Shamir Adleman
SAT	SIM Application Toolkit
SATSA	Security and Trust Services API
SIM	Subscriber Identity Module (a tamper-resistant card holding a secret key shared with the home network)
SMS	Short Message Services
SOAP	Simple Object Access Protocol
TBS	To Be Signed
TLP224	Transport Layer Protocol 224
TTP	Trusted Third Party
URL	Universal Resource Locator
USSD	Unstructured Supplementary Service Data
UMTS	Universal Mobile Telecommunication Standard
WAP	Wireless Application Protocol
WIM	Wireless Identity Module
WPKI	Wireless PKI
WSDL	Web Service Definition Language
WSIM	Wireless SIM
WTK	Wireless Toolkit (a Sun Microsystems' development platform for J2ME)
XML	Extensible Markup Language

References

[1] Mobey Forum, White Paper on Mobile Financial Services, V.1.1, Helsinki, Finland. <http://www.mobeyforum.org>, 2003 (accessed 21.02.07).

[2] Information for Development Program, Micro-payment systems and their application to mobile networks, An infoDev Report, The International Bank for Reconstruction and Development/The World Bank, Washington, DC, USA. January 2006 (accessed 21.02.07).

[3] K. Taga, J. Karlsson, Global M-Payment Report 2004. Making M-Payment a Reality, The Strategy Consultancy Arthur D. Little, Vienna, Austria, July 2004.

[4] A. Herzberg, Payments and banking with mobile personal devices, Commun. ACM 46 (5) (2003) 53–58.

[5] N. Mallat, M. Rossi, V.K. Tuunainen, Mobile banking services, Commun. ACM 47 (5) (2004) 42–46.

[6] S. Karnouskos, A. Hondroudaki, A. Vilmos, B. Csik, Security, trust and privacy in the secure mobile payment service, in: Proceedings of the 3rd International Conference on Mobile Business, New York, NY, USA, 2004.

[7] K. Poutsttchi, Conditions for acceptance and usage of mobile payment procedures, in: G.M. Giaglis, H. Werthner, V. Tschammer, K.A. Froeschl (Eds.), Proceedings of the 2nd International Conference on Mobile Business, Vienna, Austria, 2003, pp. 201–210.

[8] K. Linck, K. Pousttchi, D.G. Wiedemann, Security issues in mobile payment from the customer viewpoint, in: J. Ljungberg (Ed.), Proceedings of the 14th European Conference on Information Systems (ECIS), Göteborg, Sweden, 2006.

[9] N. Mallat, V.K. Tuunainen, Merchant adoption of mobile payment systems, in: Proceedings of the International Conference on Mobile Business (ICMB'05), IEEE Computer Society, Washington, DC, USA, 2005, pp. 347–353.

[10] S.K. Misra, N. Wickamasinghe, Security of a mobile transaction: A trust model, Electronic Commerce Research 4 (4) (2004) 359–372.

[11] Mobile Electronic Transactions Ltd., MeT Core Specification Version 1.2, Helsinki, Finland. <http://www.mobiletransaction.org/>, November 2002 (accessed 22.02.07).

[12] 3rd Generation Partnership Project, 3GPP TS 11.14: Specification of the SIM Application Toolkit (SAT) for the Subscriber Identity Module – Mobile Equipment (SIM-ME) interface, V8.17.0,

Valbonne, France. <http://www.3gpp.org/ftp/Specs/html-info/1114.htm>, September 2004 (accessed 22.02.07).

[13] M. Myers, A. Malpani, S. Galperin, C. Adams, X.509 internet public key infrastructure online certificate status protocol – OCSP, Network Working Group, Request for Comments 2560. <http://tools.ietf.org/html/rfc2560>, June 1999 (accessed 22.02.07).

[14] J. Ellis, M. Young, J2ME Web Services 1.0, Sun Microsystems, Inc., Santa Clara, CA, USA. <http://www.jcp.org/en/jsr/detail?id=172>, October 2003 (accessed 22.02.07).

[15] Interoperable Delivery of European eGovernment Services to Public Administrations, Businesses and Citizens (IDABC), IDABC Work Programme 2005–2009, third revision, Brussels, Belgium. <http://ec.europa.eu/idabc/servlets/Doc?id=25302>, May 2006 (accessed 23.02.07).

[16] N. Kreyer, K. Pousttchi, K. Turowski, Characteristics of mobile payment procedures, in: Z. Maamar, W. Mansoor, W.-J. van den Heuvel (Eds.), First International Workshop on M-Services – Concepts, Approaches, and Tools, June 26, 2002, CEUR Workshop Proceedings, vol. 61, Sun SITE Central Europe (CEUR), Lyon, France, 2002.

[17] D. McKitterick, J. Dowling, State of the art review of mobile payment technology, Technical Report. TCD-CS-2003-24, Department of Computer Science, Trinity College, Dublin, 2003.

[18] S. Karnouskos, Mobile payment: a journey through existing procedures and standardization initiatives, IEEE Communication Surveys 6 (4) (2004) 44–66.

[19] J. Ondrus, Y. Pigneur, A disruption analysis in the mobile payment market, in: R. Sprague (Ed.), Proceedings of the 38th Hawaii International Conference on System Sciences, Kona, HI, IEEE Computing Society Press, Los Alamitos, CA, 2005.

[20] Mobile Payment Forum, Risks and threats analysis and security best practices. Mobile 2-way Messaging Systems, Version 1.0, Wakefield, MA, USA. <http://www.mobilepaymentforum.org>, December 2002 (accessed 23.02.07).

[21] K. Boman, G. Horn, P. Howard, V. Niemi, UMTS security, Electronics & Communication Engineering Journal 14 (5) (2002) 191–204.

[22] V. Niemi, K. Nyberg, UMTS Security, John Wiley & Sons, Chichester, England, 2003.

[23] Finextra.com, Nordea Leads Dual-chip Mobile Payment Trials. <http://www.finextra.com/fullstory.asp?id=3220>, 24 September 2001 (accessed 23.02.07).

[24] B. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, second ed., John Wiley & Sons, New York, NY, 1996.

[25] Population Register Centre of Finland, What is the citizen certificate? Helsinki, Finland. <http://www.fineid.fi/vrk/fineid/home.nsf/en/products>, 5 August 2005 (accessed 23.02.07).

[26] Bluetooth Special Interest Group, Specification of the Bluetooth System, v. 2.0 + EDR. <https://www.bluetooth.org>, 4 November 2004 (accessed 23.02.07).

[27] L. Gong, G. Ellison, M. Dageforde, Inside Java™ 2 Platform Security: Architecture, API Design, and Implementation, 2nd ed., Addison-Wesley, Boston, MA, 2003.

[28] C. Collberg, C. Thomborson, Watermarking, tamper-proofing, and obfuscation – tools for software protection, IEEE Transactions on Software Engineering 28 (2002) 735–746.

[29] Java Community Process, Security and Trust Services API (SATSA) for Java™ 2 Platform, Micro Edition, v. 1.0, Sun Microsystems, Inc., Santa Clara, CA, USA. <http://www.jcp.org/en/jsr/detail?id=177>, 17 July 2004 (accessed 23.02.07).

[30] S. Kent, Privacy enhancement for internet electronic mail: Part II: Certificate-based key management, Network Working Group, Request for Comments 1422. <http://tools.ietf.org/html/rfc1422>, February 1993 (accessed 01.03.07).

[31] Population Register Centre of Finland, FINEID S1 – Electronic ID Application, v2.1, Helsinki, Finland. <http://www.fineid.fi/>, 15 March 2004 (accessed 01.03.07).

[32] K.D. Vriendt, eID Interoperability actions in the context of the eGovernment Action Plan, Presentation at the 10th Porvoo Group meeting, Porvoo, Finland. <http://porvoo10.net/p10/4_KDV-Porvoo-eIDatEC.ppt>, 2 November 2006 (accessed 01.03.07).

[33] Liberty Alliance Project, Tier 2 Business Guidelines: Mobile Deployments, <http://www.projectliberty.org/>, 6 February 2004 (accessed 03.03.07).

[34] A. Vilmos, S. Karnouskos, SEMOPS: design of a new payment service, in: V. Mařík, W. Retschitzegger, O. Štěpánková (Eds.), Proceedings of the 14th International Conference on Database and Expert Systems Applications, Lecture Notes in Computer Science, vol. 2736, Springer, Prague, Czech Republic, 2003, pp. 865–869.