Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 30 (2007) 1697–1712
computer communications
www.elsevier.com/locate/comcom

Improving authentication of remote card transactions with mobile personal trusted devices

Andrea Bottoni, Gianluca Dini *

Dipartimento di Ingegneria dell'Informazione: Elettronica, Informatica, Telecomunicazioni, University of Pisa, Via Diotisalvi 2, 56100 Pisa, Italy

Received 21 May 2006; received in revised form 5 February 2007; accepted 6 February 2007
Available online 16 February 2007

Abstract

Credit card transactions are a popular and diffused means of payment over the network. Unfortunately, current technology does not allow us to technically solve disputes that may arise in such transactions. Thus these disputes are often solved on legal and administrative basis. In these cases, responsibility is not necessarily allocated fairly and the problems of managing the resulting risks have proven to be an impediment to the growth of electronic commerce.

In this paper we present a protocol for credit card transactions over the network that uses personal trusted devices (e.g., a cellphone or a PDA) to improve the technical management of disputes and permit a more fairly allocation of risks between customer and merchant. The protocol also defines a practical trade off between the security properties of these devices and the resource limitations deriving from their form factor. Furthermore, by means of formal methods, we specify the security requirements of a personal trusted device and analyse the security properties of the protocol. Finally, we argue that a cellphone practically fulfills the above security requirements and thus can be used as a personal trusted device.

Keywords: Security; Authentication; e-Payment; Trusted device

1. Introduction

Remote card transactions are credit card transactions in which cardholder and merchant "meet over the network". Remote card transactions are a popular and diffused means of payment over the network [54].

In remote card transactions, the incidence of risk is quite different from that where the card is presented by the customer to the merchant. Bohm et al. provided a very deep and precise analysis [7]. Briefly, in a remote card transaction no voucher is signed and the customer only provides the merchant with information apparent from the face of the card. The ability to provide the card information does not depend on the possession of the card. In fact, such an information is available to anyone through whose hands the card has passed during earlier transactions. It follows that there is very little impediment to fraud either by the cardholder falsely repudiating a genuine transaction or by an impostor using the card details without authority. If the cardholder repudiates a remote card transaction, the bank has no basis on which to charge the cardholder's account. Faced with apparently unmanageable risks of this kind, banks have adopted the approach requiring the merchant to carry the risk. Thus, if the cardholder repudiates a remote card transaction for which there is no voucher signed by the cardholder, the bank makes a "chargeback", i.e., obtains reimbursement from the merchant of anything paid to the merchant in respect of the transaction. The merchant is in practice unable to transfer the risk to anyone else, since he is unlikely to be able to prove who initiated the transaction.

Although simple, this liability regime has important implications. The greatest risk to the merchant arises from the provision of online services. Although online services

* Corresponding author. Tel.: +39 050 2217 549; fax: +39 050 2217 600.
E-mail addresses: a.bottoni@iet.unipi.it (A. Bottoni), g.dini@iet.unipi.it (G. Dini).

0140-3664/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2007.02.006

have been provided for long, they have expanded greatly with the commercialization of the Internet. Provision of online services is one of the most effective uses of the Internet for electronic commerce. Small and medium enterprises are among those which can derive the greatest benefit from access over the Internet, but can least afford exposure to the risks which remote card transactions place on merchants. Therefore, the problem of managing the resulting risks for the merchant may well prove to be an impediment to the growth of electronic commerce in online services.

Technological solutions have been proposed to improve the security of remote card transactions. The most relevant are *Secure Socket Layer*, SSL [27], and *Secure Electronic Transactions*, SET [45]. In remote card transactions carried out using a web browser to connect to the merchant, it is possible to establish a secure connection so that the information is delivered in encrypted form using protocols such as SSL or TLS [13]. This procedure is widely followed and provides protection against interception of the card information in transit. However, it cannot affect the widely availability of the card information from other sources, and cannot provide evidence that the supplier of the card information is authorised by the cardholder. Thus it does not materially reduce the merchant's risk [7,54].

Secure Electronic Transactions (SET) is a standard promulgated by Visa and Mastercard. SET allows a merchant to check whether the bank will accept the cardholder's authority as genuine. The intent is to remove the risk from the merchant, or at least reduce it. SET has not gained acceptance perhaps because it is over elaborate and its implementation is burdensome and expensive [54]. Apart from that, there is a subtle point about its security model that is central to this paper. SET improves the merchant's exposure to risk of chargeback by precluding a cardholder from repudiating a SET transaction which appears to have been authorised by that cardholder. However, this gives rise to an unacceptable shift of the risk from the merchant to the customer. In fact, the risk of the customer of losing control of the means of authorising a SET transaction, namely information stored in electronic form, is very different from the risk of losing a plastic card. The current version of SET was designed for common desktop PCs as typical user terminals, and with the Internet as the transport network. PCs are unlikely to meet any serious security requirement for several reasons [7,41]. In such an environment, the customer is exposed to the risk of his private key being compromised without the means of detecting the compromise until the fraudulent use becomes evident. A sophisticated attack might leave no evidence and the customer would be thus left in a weak position to resist an assertion of the bank that the remote card transaction was correctly authorised.

In this paper, we present a protocol to improve authentication of remote card transactions by means of personal trusted devices. The main objective of the protocol is to shift the risk to a more balanced position between the merchant and the customer. Improving authentication consists in providing non-repudiable proof of transaction authorization both from the customer and the merchant. These proofs make it possible to improve the technical solution of disputes. In the authorization process, the personal trusted device plays a crucial role as it allows the customer to generate his strong proof and, at the same time, reduces the risk that he can lose control of the means of authorising a payment transaction.

More in detail, the paper makes the following contributions.

- First, it shows that the overall security of an electronic payment system can be greatly increased by means of a personal trusted device. In particular, the use of such a kind of device makes it possible to improve the way to solve disputes in a technical way.
- Second, the electronic payment protocol takes into account both the security limitations of conventional user computers (e.g., home/office PCs) and the resource limitations of personal trusted devices (small screen and keyboard), and defines a practical trade-off between security and usability. A PC is used to browse and select goods, whereas a personal device is used to authorize a payment transaction. In order to issue such an authorization the customer has only to read a few information items on the device screen, enter a PIN, and press just a few buttons.
- Third, by using a formal method, namely an extended version of the BAN logic [9,1], we state the trust requirements of the personal device. So far, these requirements have been informally stated [40]. To the best of our knowledge, ours is the first effort to formalize them.
- Fourth, we exploit the formal framework to highlight the security limitations of a conventional, ''open'', PC-based system, with or without smart cards, and to argue that a GSM/UMTS cellphone can be practically considered a personal trusted device as long as it is part of the ''closed'' GSM/UMTS application framework. The use of cellphones in e-commerce has been suggested by many [5,10,12,26,33–35,44,46,49,50]. In this paper we give a theoretical and architectural basis to this statement.

The paper is organized as follows. In Section 2, we briefly introduce the payment model based on credit cards. The proposed electronic payment system has the objective to interface with the pre-existing credit card payment systems without changing them. In Section 3, we specify the basic security requirements the proposed electronic payment system is required to fulfill. In Section 4, we present the electronic payment system. In Section 5, we make a security analysis of the proposed payment protocol. In this activity we will use a variation of the BAN logic [9]. In Section 6, we discuss the electronic payment protocol. Finally, in Section 7, we make conclusive remarks.

2. The electronic payment model

In an electronic commerce (e-commerce) system customers and merchants exchange money for goods and services, whereas financial institutions are responsible to link "bits" to "money" and give support to the actual flow of money from the customer to the merchant [2].

The electronic payment protocol we are going to present is based on the existing credit-card payment system which, in short, is structured as follows. With reference to Fig. 1, a *payment system provider* operates the credit-card payment system and is responsible for the authorization and clearing of credit card transactions. The payment system provider maintains a fixed business relationship with a number of financial institutions which play two roles: the *issuer* and the *acquirer*. Every merchant establishes a contractual relationship with an acquirer which provides payment processing services in return for a percentage fee. Every customer has a contractual relationship with an issuer: the customer receives a credit card and promises to pay a minimal yearly fee and stay within a credit limit. In return for this, the issuer is the ultimate responsible for the payment of the customer's debt.

With reference to this payment system, the customer and the merchant initially reach an agreement on the service/goods to purchase. Then, the customer sends a payment order, a signed form that explicitly authorizes the payment transaction. The merchant attaches the payment order to a payment request and sends all to the payment system provider to obtain on-line authorization of the payment. If the payment system provider grants the authorization, the merchant provides the required service/goods. At this point, the merchant is allowed to come back to the acquirer and reclaim the real payment. The gateway sends a clearing request to the payment system provider which arranges for the transaction clearing.

In this paper we deal with the authorization phases of the payment protocol. Therefore, with reference to Fig. 1, we consider the interactions marked as "Payment order" and "Payment request, Payment order," that involve only the customer, the merchant, and the acquirer. As to the acquirer, we only focus on its role of *gateway* between the electronic payment protocol and the pre-existing financial network which thus can remain unchanged.

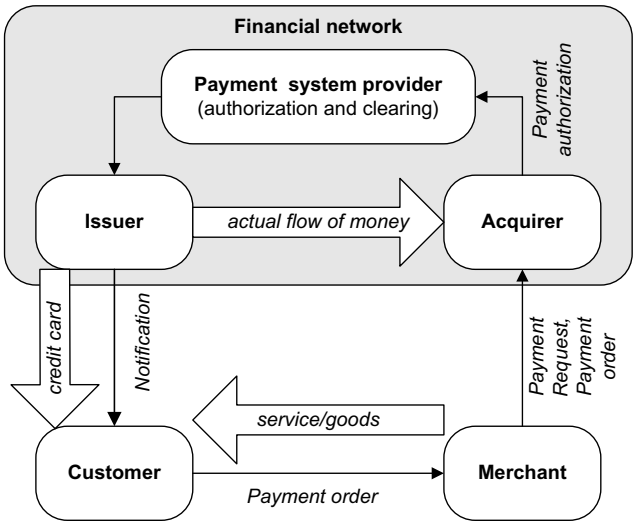

Fig. 1. Generic model of a credit-card payment system.

We assume that each customer receives his credit card from an issuer. Every credit card has a unique credit card number and an expiration date both known to the payment system provider. Credit card numbers follow the standard specifications and convey usual information about issuers and account numbers.

3. Requirements

With reference to the payment model in Section 2, we consider an electronic commerce system in which a purchase takes place in two phases: negotiation and payment. In the *negotiation phase*, the customer browses a merchant's site and selects the desired items to buy. During this phase, the merchant and the customer reach an agreement upon a set of information items that describe the purchase. These information items constitute the *order instructions* and comprise the description and price of items to buy, the time of the purchase, the currency, the modalities of shipping, and so forth.

After completion of the negotiation phase, the *payment phase* begins during which the electronic payment actually takes place. In this phase, the customer sends the acquirer, through the merchant, a *payment order*, to explicitly authorize the payment. The payment order specifies both the order instructions and the *payment instructions* which enclose the credit card number and the expiration date. In its turn, the merchant sends the acquirer a *payment request* to authorize the payment. The payment request specifies the order instructions. Upon receiving the payment request from the merchant and the payment order from the customer, the acquirer first ascertains whether they reflect the same purchase and, if this is indeed the case, generates an on-line *authorization request* to the financial network. Upon receiving a response from the financial network, the acquirer returns an *authorization response* to both customer and merchant.

An electronic payment system must meet the integrity security requirement. In general, integrity refers to the ability to detect and/or prevent improper or unauthorized modifications to the system state. In an electronic payment system, integrity can be phrased in terms of preventing any amount of money from being debited or credited without the user explicit consent.

With reference to the payment model described above, integrity requires that when the acquirer requests a payment authorization aimed at debiting a customer's account and, correspondingly, crediting to a merchant's account, the acquirer must be in possession of the proof that the customer has explicitly ordered the payment, as well as the proof that the merchant has explicitly requested the

payment be made to him. This requires the acquirer to be able to authenticate the origin of both the payment order and the payment request. So doing, as data origin authentication includes data integrity [37], the acquirer is also ensured that the payment order and the payment request have not been altered in an unauthorized way since the time they were created. However, in an electronic commerce context, this is not sufficient. It is also necessary to guarantee the timeliness and uniqueness of a payment transaction as well as its non-repudiability. While the former prevents undetectable payment order/request replays, the latter allows an unbiased third party to equitably resolve disputes. For this reason, we require that each proof is also *strong*, i.e., it cannot be either replayed or repudiated. If we call *customer authorization* a strong proof of payment order from the customer and *merchant authorization* the corresponding strong proof of payment request from the merchant, then

Requirement 1 (*Authorization*)**.** The acquirer requests a payment authorization to the financial network if and only if it holds both customer and merchant authorizations.

Both the merchant and the customer need a strong proof of whether the payment has been authorized or not. If we call *acquirer response* a strong proof of the authorization response from the acquirer, then

Requirement 2 (*Response*)**.** For any given authorization request, the merchant and the customer need to have the corresponding acquirer response.

In addition to integrity, the electronic payment system has to satisfy the secrecy requirements. In general, secrecy refers to the ability to keep the content of information from all but those authorized to have it. In an electronic payment system, secrecy can be phrased in terms of preventing any non-authorized entity to see the order and payment instructions.

Requirement 3 (*Payment Secrecy*)**.** No one but the customer and the acquirer can know the payment instructions of any given purchase.

Requirement 4 (*Order Secrecy*)**.** No one but the merchant and the customer can know the order instructions of any given purchase.

The security requirements for an electronic payment system may vary, depending on the system features and the trust assumptions placed on the system itself. For instance, anonymity is often required [2]. However, we believe the above requirements are unrenounceable for any remote card transaction system.

For a practical deployment of an electronic payment system, other requirements are crucial [6]. For instance, an electronic payment system is certainly expected to meet the availability and reliability requirements. While availability refers to the ability to make and receive payments as desired, reliability refers to the ability to preserve isolation, durability and consistency of payment transactions in spite of a network or system crash [2]. In order to fulfill the availability and reliability requirements, standard mechanisms and methodologies can be applied [31,36]. For instance, electronic payment transactions are typically implemented as atomic transaction in order to fulfill the ACID properties dictated by the reliability requirement [32]. However, this topic is largely independent of the paper topic, and therefore we shall not treat it here any longer.

4. The payment service

4.1. The system model

With reference to Fig. 2, in the negotiation phase, a customer accesses a merchant's web site from a *host node*. This is a personal computer with standard hardware platform, operating system and application software, and is equipped with plentiful of resources for computing, storage, graphical rendering and communication. Communication between the host node and the merchant can be protected using customary means such as SSL [27] or TLS [13], for example. The host node may be the customer's home computer as well as any other personal computer placed in virtually any location that is accessible in the Internet, e.g., the customer's office, an Internet Café and so on. Thus the host node allows the customer to comfortably browse the Web but it cannot be considered a trustworthy device [41]. For this reason, we assume that the host node assists the customer during the negotiation phase only.

As soon as the customer has terminated the negotiation phase, the customer clicks on the "payment button" to

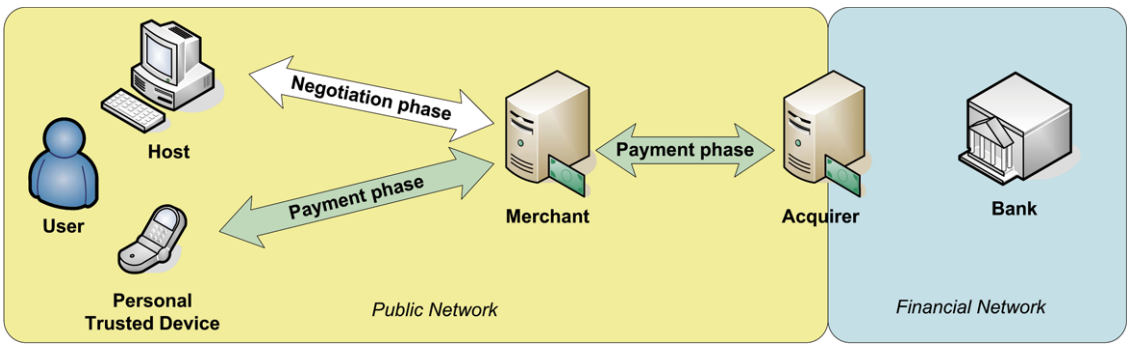

Fig. 2. The system model.

terminate that phase and begin with the payment one. In this phase the *payment protocol* is performed. The execution of the protocol involves the acquirer, the merchant and the customer. During the execution of the protocol, the customer is assisted by his own *personal trusted device*, a trustworthy mobile user device the customer uses to control all security relevant and legally significant actions of the payment.

The personal trusted device satisfies certain trust assumptions that will be formally discussed in Section 5. For the moment, we informally assume that the personal trusted device provides both the *personal-agent trust* and the *captured-agent trust* type of trustworthiness [40]. The personal-agent trust implies that the personal trusted device acts according to the user's wishes while it is in the user's hands. For instance, it does not authorize unintended payment transactions. The captured-agent trust implies that the personal trusted device protects the user even while it is not in the user's hands. For instance, if the device is lost, stolen or given away (e.g., for maintenance), no one can authorize payment transactions in the legitimate user's name.

In order to provide these forms of trust, we informally assume that the personal trusted device is equipped with a tamper-resistant security module (e.g., a smart card), is not subject to the fake-terminal attack, and requires user identification for each security-critical command. Furthermore, we assume that the personal trusted device is personalized with a private key trusted by the user to digitally sign payment orders, and with a non-repudiation PIN the customer enters to generate a signed statement authorizing a payment transaction. Both the signing key and the non-repudiation PIN are stored in the tamper resistant module to physically protect them from unauthorized accesses. Finally, we assume that production, distribution, and personalization of the device is securely done off-line with well-known means [40].

We assume that end-to-end communication between the merchant and the personal trusted device is possible. Abstracting away from the network level, such a communication can take place through TCP/IP, WAP [24,25] or even SMS [15]. As it will appear clearer in Section 5, we assume that it is possible to establish an end-to-end secure channel between the merchant and the personal trusted device for secrecy only [27,53].

4.1.1. Cryptography, keys and certificates

Our protocols make use of cryptography. We denote by $\{x\}_K$ the encryption of quantity x by means of key K. Whether the encryption is under a symmetric or a public-key algorithm will be clear from the context. We assume that each principal P has a pair of private and public keys that we denote by K_p^{-1} and K_p, respectively. We denote by $\{x\}_{K_p^{-1}}$ the digital signature of P for x. A prudent encryption engineering suggests that the key-pair used for digital signatures should be distinct from that used for encryption. However, for the sake of conciseness, this distinction is not reflected in the protocol description below.

In order to guarantee the authenticity and validity of public keys, we use *certificates*. A certificate is a data structure composed of two parts: a *data part* and a *signature part* [37]. The data part contains cleartext data including, as a minimum, a public key, a string indentifying the principal to be associated therewith, and a validity period. The signature part contains the digital signature of a *Certification Authority* (CA) over the data part, thereby binding the principal's identity to the specified public key. The Certification Authority is a trusted third party whose signature on the certificate vouches for the authenticity of the public key bound to the principal identity. Anyone with the public key of this authority can verify this assertion and, providing he trusts the authority, he uses the indicated key to authenticate the indicated principal.

For simplicity, but without lack of generality, we assume a single certification authority CA, trusted by all principals, and whose public–private key pair is denoted by (K_{ca}, K_{ca}^{-1}). The key K_{ca} must be distributed in an authenticated manner to every principal. We assume this is done off-line by means of well-known methods [42].

Table 1 lists principals' keys and related certificates issued by the Certification Authority CA. As to certificate $Cert_a$, we assume that Merchant M holds it as a consequence of the business relationships established with the Acquirer. Furthermore, certificate $Cert_d$ slightly differs from the other certificates. The additional field U specifies the customer to whom the device is assigned. Intuitively, this certificate states that device D is trusted to "speak for" customer U. In other words, every message sent by D can be considered "as if" it were sent by the customer U. In Section 5, we shall deepen this point in a more formal way.

Finally, we denote by h a collision-resistant hash function (CRHF). This is a hash function with the following properties: a digest $h(x)$ for any input quantity x can be computed efficiently, but it is computationally infeasible to find an x' given x such that $h(x) = h(x')$ (*second-preimage resistance*), or to find a pair (x, x') such that $h(x) = h(x')$ (*collision resistance*).

A further property that we require to the hash function is that it is computationally infeasible to compute x given $h(x)$ (*preimage resistance*). Although in theory collision resistance does not guarantee preimage resistance, in practice a CRHF has always the additional property of preimage resistance [37]. Thus, in the following we assume that the CRHF satisfies all the three properties.

Table 1
Principals, keys and related certificates

Principal	Key pair	Certificate
Acquirer A	(K_a, K_a^{-1})	$Cert_a = \{A, K_a, T_a\}_{K_{ca}^{-1}}$
Merchant M	(K_m, K_m^{-1})	$Cert_m = \{M, K_m, T_m\}_{K_{ca}^{-1}}$
Personal trusted device D	(K_d, K_d^{-1})	$Cert_d = \{D, U, K_d, T_d\}_{K_{ca}^{-1}}$

Quantities T_a, T_m, and T_d specify the validity period of the corresponding certificates.

4.2. The payment phase

4.2.1. Acquirer's data structures

In order to support the payment phase the acquirer is equipped with stable storage where it maintains the following data structures:

- The *Pending Transaction List*, *PTL*, which specifies the *pending transactions*, i.e., the payment transactions whose execution has started but has not completed yet. Conceptually, PTL has one element for each pending transaction. The element is inserted in the list when the transaction execution begins, and is extracted from the list when the execution terminates. Initially, the list is empty.
- The *Completed Transaction List*, *CTL*, which specifies the *completed transactions*, i.e., the payment transactions whose execution has completed but that has not been yet reported in the next statement of account. Conceptually, CTL has one element for each completed transaction that specifies whether the transaction has completed successfully or not. The element is inserted when the transaction execution terminates and is removed when the transaction is reported in the next statement of account.

4.2.2. The payment protocol

The customer carries out the negotiation phase through the host node, but he carries out the payment phase through his personal trusted device. It follows that, before the execution of the payment protocol can be started, the customer's device has to be taken into the proper initial state. For instance, the personal trusted device has to receive the order instructions, in order to digitally sign them and thus authorize the transaction, as well as receive the acquirer's certificate. A *prologue*, whose execution precedes that of the payment protocol, has been conceived just with this aim.

Let *OI* denote the order instructions. The prologue is composed of the following actions:

1. Initially, the merchant generates a random number ρ, and then asks the acquirer for a unique transaction identifier.
2. Upon receiving this request, the acquirer generates a *unique transaction indentifier*, n_a, inserts it into the Pending Transaction List PTL, and returns it to the merchant.
3. Upon receiving the transaction identifier, the merchant sends the four tuple (n_a, ρ, *OI*, $Cert_a$) to the customer's personal trusted device.
4. Upon receiving the tuple, the customer's personal trusted device displays the merchant identifier *M* and the order instructions *OI* on its output device O_d.
5. Upon viewing these quantities on the screen, the customer ascertains that they correctly describe the agreed purchase. If this is indeed the case, the prologue execution completes successfully.

Upon successful completion of the prologue, the execution of the payment protocol may begin. With reference to Fig. 3, the payment protocol consists of the following actions:

1. The customer explicitly authorizes the transaction by entering the non-repudiation PIN, *NRP*, into his own personal trusted device (message M1).
2. Upon receiving the non-repudiation PIN from its input device I_d, the personal trusted device verifies it, generates a nonce n_d, computes $EPI = \{PI\}_{K_a}$ by encrypting the payment instructions *PI* with the acquirer's public key, computes $HOI_u = h(OI \| \rho)$, prepares a message containing the *payment order* (message M2), and sends it to the merchant.
3. Upon receiving the payment order message from the customer, the merchant generates a nonce n_m, computes $HOI_m = h(OI \| \rho)$, builds a *payment request*, attaches it to the just received payment order, and, finally, sends the resulting message to the acquirer (Message M3).
4. Upon receiving the message containing the payment order and request from the merchant, the acquirer *A* performs the following actions:
   (a) Initially, the acquirer performs the following checks:
      i. the acquirer verifies the digital signatures on both the payment order and the payment request in order to ascertain their authenticity;
      ii. then, the acquirer ascertains that neither the payment order nor the payment request have been replayed by checking that the values of their respective n_a fields are equal to each other, and that such a value is present in the Pending Transaction List PTL; and,
      iii. finally, the acquirer verifies that HOI_u is equal to HOI_m in order to ascertain that merchant and customer agree on the purchase.

   (b) Then, if all the above checks are successful, the acquirer decrypts the quantity *EPI* to obtain the payment instructions *PI* and goes through the financial network to obtain the payment authorization.
   (c) Finally, upon receiving a response *R*, the acquirer removes n_a from the Pending Transaction List PTL, records the pair (n_a, *R*) in the Completed Transaction List CTL, and returns the merchant a *response message* containing the response *R* (message M4).

$$
\begin{array}{lll}
\text{M1} & U \to D: & NRP \ \textbf{on} \ I_d \\
\text{M2} & D \to M: & \{\text{PayOrd}, M, U, n_a, n_d, HOI_u, EPI\}_{K_d^{-1}}, Cert_d \\
\text{M3} & M \to A: & \{\text{PayReq}, M, U, n_a, n_m, HOI_m\}_{K_m^{-1}}, Cert_m, \\
 & & \{\text{PayOrd}, M, U, n_a, n_d, HOI_u, EPI\}_{K_d^{-1}}, Cert_d \\
\text{M4} & A \to M: & \{\text{PayResp}, A, M, U, n_m, n_d, R\}_{K_a^{-1}}, Cert_a \\
\text{M5} & M \to D: & \{\text{PayResp}, A, M, U, n_m, n_d, R\}_{K_a^{-1}}, Cert_a \\
\text{M6} & D \to U: & A, M, OI_u, R \ \textbf{on} \ O_d
\end{array}
$$

Fig. 3. The payment protocol.

5. Upon receiving the response message from the acquirer, merchant M checks the digital signature to ascertain that it comes from the acquirer A, verifies that the message contains the fresh quantity n_m to ascertain that the message is not a replay, and, then, forwards it to the personal trusted device (message M5).
6. Upon receiving the response message from the merchant, the personal device D checks the digital signature to ascertain that it comes from the acquirer A, verifies that the message contains the fresh quantity n_d to ascertain that the message is not a replay, and, then, displays it to the customer (message M6).

The merchant may fail to send the customer the payment response message, and therefore the customer would not be informed about the outcome of the payment transaction. However, the customer may obtain such an information by directly asking it to the acquirer. In practice, the customer sends the acquirer a signed *inquiry* message conveying the unique transaction identifier n_a. Upon receiving this message and successfully verifying the digital signature, the acquirer accesses the Completed Transactions List, CTL, using n_a as a key, retrieves quantity R and returns it to the customer.

5. Security analysis

In this section we argue that our protocol meets the requirements mentioned in Section 3. More precisely, in Section 5.1, we argue that the protocol fulfills the secrecy requirements whereas in Section 5.2 we consider the integrity requirements. All these arguments are informal and are not intended to constitute a rigorous proof of security.

5.1. Analysis of secrecy requirements

Proposition 1. No one but the customer and the acquirer can know the payment instructions of any given purchase.

Proof. In message M2, the customer U sends the acquirer A the payment instructions PI encrypted under the acquirer's public key K_a. It follows that only the acquirer can decrypt this encrypted material and thus retrieve the payment instructions related to the customer (step 4a of the payment protocol). □

According to Requirement 1, the successful execution of the payment protocol requires that the acquirer ascertains that customer and merchant agree upon the purchase. A possible approach would be to let the acquirer check that the order instructions in the user's and merchant's hands, respectively, are actually the same. However, this would contrast with Requirement 4 because the acquirer would discover the order instructions. In order to solve this problem, we let the acquirer check that the digests of the order instructions, namely HOI_u and HOI_m, respectively, are equal (step 4(a)iii), and assume that customer and merchant agree upon the purchase if HOI_u and HOI_m are equal. The rationale is that, according to the collision-resistance property of the hash function h (Section 4.1.1), if two quantities produce the same digest then they can be considered equal for any practical purpose.

Proposition 2. No one but the merchant and the customer can know the order instructions of any given purchase.

Proof. During the payment protocol, the merchant and the customer's personal trusted device transmit the hash code of the order instructions (steps 2 and 3). According to the preimage resistance property of the hash function, it is not computationally feasible to determine the order instructions from the hash code. However, if the set of possible order instructions has a reduced number of elements, a dictionary attack (exhaustive analysis of data) would be possible hence violating Requirement 4. The random number ρ is used to salt the order instructions and thus avoid this kind of attack [37]. □

As it turns out from Proposition 2, the payment protocol does not violate the Order Secrecy Requirement. However, the order secrecy can be violated in other moments of the purchase. In the negotiation phase, the order instructions are exchanged between the host node and the merchant, whereas in the prologue, the merchant transmits the order instructions and the salt to the personal trusted device. It follows that the secrecy of the order instructions must be protected also during these transmissions. To this purpose, SSL [27] or TLS [13] can be used to secure communication between host and merchant in the negotiation phase.

Analogously, the prologue requires secure end-to-end communication between the merchant and the personal trusted device. The fulfillment of this requirement depends upon the chosen technology for the personal trusted device. For example, if the personal trusted device is a GSM/UMTS cellphone, secure end-to-end communication can be established between an application on the SIM and a service provider on the Internet by means of secured packets implemented over USSD or SMS [17]. Alternatively, end-to-end secure communication can be provided by means of WAP [52]. While WAP architecture was used to suffer from man-in-the-middle, now it has evolved into a more secure one that provides end-to-end security [53]. As a further example, if the personal trusted device is a PDA or a laptop, it can access the network according to IEEE 802.11 and then establish secure end-to-end communication by means of SSL or TLS.

5.2. Analysis of integrity requirements

As to the integrity requirements, we analyse the protocol with the BAN logic [9] enriched with simple extensions to handle secure and timely channels [1]. Appendix A concisely reports the statements and postulates of the extended

logic that we need in this paper. For a thorough treatment of the logic, we refer readers to the original papers.

In brief, the BAN logic allows us to describe the beliefs of the principals in the course of authentication. The underlying idea of the logic is that a principal considers authentic a message if it is encrypted with a relevant key and is fresh, i.e., it has been generated in the current execution instance of the protocol. Other basic assumptions are that principals assert only statements in which they believe, and that certain principals are considered authorities on certain types of statements.

Abadi et al. have extended the BAN logic to capture the presence of secure and timely channels. Furthermore, they make a distinction between input and output channels because a device may have a channel of one type but not of the other. The underlying idea of the extension is that a principal considers authentic a message if it is transmitted over a secure and timely channel.

In the following we proceed as usual in the BAN logic. First, we specify the properties the protocol has to achieve and the assumptions under which the protocol works. Such specification are formula constructed by means of the constructs reported in Appendix A. Then, we provide an idealized version of the protocol and show that it achieves the specified properties from the specified assumptions. We do that by transforming each protocol step in an idealized form where a message is a formula. Then, we annotate each protocol step with logical formulas. The main rule for deriving legal annotations consists in applying the logical postulates in Appendix A. For each protocol step, we apply logical postulates to the set of formulas composed of those holding before the message and those contained in the message itself and derive the set of formulas holding afterwards. So doing, step by step, we can follow the evolution from assumptions to properties.

The following set of properties formalize Requirements 1 and 2. In particular, Properties P1–P2 formalize Requirement 1 and Properties P3–P4 formalize Requirement 2.

P1. A **believes** (U, M, OI_u). Acquirer A believes that the customer U has ordered a payment in favour of merchant M for a purchase specified by OI_u.

P2. A **believes** (M, U, OI_m). Acquirer A believes that merchant M has requested a payment for a purchase made by customer U and specified by OI_m.

P3. M **believes** (R, A, U, M, OI_m). The merchant M believes that R is the response from the acquirer A to a payment requested to customer U for the purchase specified by OI_m.

P4. U **believes** (R, A, U, M, OI_u). The customer U believes that R is the response from the acquirer A to a payment ordered in favour of merchant M for a purchase specified by OI_u.

Fig. 4 shows the idealized version of the payment protocol. For simplicity, but without lack of generality, in the idealized protocol we shall refer to *OI* rather than *HOI*. Intuitively, the rationale for this choice is that in the real protocol *HOI* is used in the place of *OI* for confidentiality purposes, whereas in the idealized version we are interested in the authentication properties of the protocol.

Furthermore, in the idealized protocol we use the operator **on** introduced by Abadi et al. to explicitly specify the secure channels on which messages are transmitted [1]. If secure channels are available, whether or not a message is transmitted on one of them matters. If a message is transmitted on a secure channel, the operator **on** allows us to specify the channel. Such a specification is necessary to correctly apply the logic postulates (Appendix A). In contrast, if a message is transmitted on an insecure channel, it is meaningless to specify the channel. Therefore, we only specify the channels for messages M1 and M6 because only these messages are transmitted on secure channels. The remaining messages (M2–M5) are transmitted through insecure channels that we do not need to specify.

In the rest of this section we shall prove that the proposed electronic payment protocol fulfils the above properties if the following set of assumptions holds. For the sake of increased readability, assumptions are grouped in four groups: assumptions about keys and secrets; assumptions about freshness; assumptions about channels; assumptions about trust.

Assumptions about keys and secrets. These assumptions specify the initial setup of keying and secret material for authentication purposes. The Certification Authority knows the public keys of every participant and each participant knows the Certification Authority's public key (A1, A2). The customer can use the non-repudiation PIN *NRP* to authenticate himself to his personal trusted device (A3).

$$
\begin{array}{lll}
\text{M1} & U \to D\text{:} & \langle M, OI_u \rangle_{NRP} \ \textbf{on}\ I_d \\
\text{M2} & D \to M\text{:} & \{U\ \textbf{believes}\ (U, M, OI_u), n_a, n_d\}_{K_d^{-1}}, \left\{\overset{K_d}{\mapsto} D, D\ \textbf{controls}\ U\ \textbf{believes}\ X, T_d\right\}_{K_{ca}^{-1}} \\
\text{M3} & M \to A\text{:} & \{M, U, OI_m, n_a, n_m\}_{K_m^{-1}}, \left\{\overset{K_m}{\mapsto} M, T_m\right\}_{K_{ca}^{-1}}, \\
 & & \{U\ \textbf{believes}\ (U, M, OI_u), n_a, n_d\}_{K_d^{-1}}, \left\{\overset{K_d}{\mapsto} D, D\ \textbf{controls}\ U\ \textbf{believes}\ X, T_d\right\}_{K_{ca}^{-1}} \\
\text{M4} & A \to M\text{:} & \{R, A, M, U, n_m, n_d\}_{K_a^{-1}}, \left\{\overset{K_a}{\mapsto} A, T_a\right\}_{K_{ca}^{-1}} \\
\text{M5} & M \to D\text{:} & \{R, A, M, U, n_m, n_d\}_{K_a^{-1}}, \left\{\overset{K_a}{\mapsto} A, T_a\right\}_{K_{ca}^{-1}} \\
\text{M6} & D \to U\text{:} & (A\ \textbf{believes}\ R)\ \textbf{on}\ O_d
\end{array}
$$

Fig. 4. The payment protocol in the idealized form. X represents any customer's belief.

A1. CA **believes** $(\forall P \in \{A, M, D\}. \overset{K_p}{\mapsto} P)$. Certification Authority CA believes that K_a, K_m, and K_d are keys to communicate with the acquirer A, the merchant M, and the device D, respectively.

A2. $\forall P \in \{A, M, D\}$. P **believes** $(\overset{K_{ca}}{\mapsto} CA)$. Acquirer A, merchant M, and device D believe that K_{ca} is the public key of the certification authority CA.

A3. D **believes** $U \overset{NRP}{\rightleftharpoons} D$. The personal trusted device D believes that NRP is a secret that will never be told to anyone but U and D.

Assumptions about freshness. Assumption A4 specifies fresh quantities. For instance, if the acquirer A sees quantity n_a in a message then the acquirer can derive that the message is not a replay. Assumption A5 specifies that certificates are within their validity period and thus are not expired.

A4. A **believes fresh** (n_a), M **believes fresh** (n_m), D **believes fresh** (n_d). Every party believes that any nonce it generates is fresh, that is, the same nonce is never used in two different execution instances of the protocol.

A5. A **believes fresh** (T_d), A **believes fresh** (T_m), M **believes fresh** (T_a), D **believes fresh** (T_a). Every party believes the needed certificates are within the validity period.

Assumptions about channels. The assumptions specify that the personal trusted device has secure input and output channels. This means that a message received by the personal trusted device (customer) on the input (output) channel has been recently transmitted by the customer (personal trusted device).

A6. U **believes** $\overset{O_d}{\prec} D$, U **believes timely** (O_d). Customer U believes that O_d is a secure and timely channel from D, that is, messages on O_d are known to have been sent by D recently.

A7. D **believes** $\overset{I_d}{\prec} U$, D **believes timely** (I_d). Device D believes that I_d is a secure and timely input channel, that is, any message on I_d is known to have been sent by U recently.

Assumptions about trust. These assumptions specify the level of trust we place in each participant. The Certification Authority is trusted to correctly certify principals (A8). Therefore, principals believe in the Certification authority when it makes statements (certificates) about the authenticity of the public keys of other principals. Customer, merchant and acquirer are trusted to correctly issue payment orders, payment requests and authorization responses, respectively (A9–A11). The personal trusted device is trusted to correctly relay messages (Assumptions A12, A13). That is, upon receiving a message, the personal trusted device forwards it to the legitimate destination without modifying it. The Certification Authority is trusted to correctly attestate this behaviour (A14).

A8. $\forall P, Q \in \{A, M, D\}$, P **believes** CA **controls** $\overset{Kq}{\mapsto} Q$. Every principal trusts the Certification Authority CA to correctly certify other principals.

A9. A **believes** (M **controls** OI_m). The acquirer trusts the merchant to control the payment request it issues.

A10. A **believes** (U **controls** OI_u). The acquirer trusts the customer to control the purchases for which he utters payment orders.

A11. M **believes** A **controls** R, U **believes** A **controls** R. M and U trust the acquirer A to generate the correct response to the payment authorization request.

A12. $\forall$ belief X, CA **believes** (D **controls** (U **believes** X)). The Certification Authority CA trusts the customer U's personal trusted device to relay the customer's beliefs.

A13. $\forall$ belief X, U **believes** (D **controls** (A **believes** X)). The customer trusts his own personal trusted device to correctly relay the beliefs of the acquirer A.

A14. $\forall$ belief X, A **believes** CA **controls** (D **controls** (U **believes** X)). The acquirer trusts the Certification Authority to properly certify the ability of the trusted device to correctly forward the customer's beliefs.

5.2.1. The protocol analyzed

With message M1, customer U tells his personal device D that he believes that OI_u correctly specifies the purchase. The customer uses the non-repudiation PIN, NRP, to convince the personal personal device D of his identity (Assumption A3). Since the device receives this message from a secure and timely channel (Assumption A7), then the device achieves the belief D **believes** U **believes** (U, M, OI_u), i.e., D believes that customer U has recently uttered a payment order in favour of Merchant M for a purchase OI_u.

With message M2, the personal trusted device forwards that belief to the merchant M together with its certificate. This certificate makes basically two statements. The first one is normally about the device's key (Assumption 1). The second one states that the device "speaks for" the customer or, in other words, that it correctly relays the customer's beliefs (Assumption 12).

Upon receiving this message, the merchant forwards its contents to the acquirer A by attaching them to message M3. With this message, the merchant sends the acquirer both his own belief about the payment request and his certificate.

The analysis of the beliefs the acquirer achieves from message M3 is divided into two parts. Initially, we analyse the beliefs deriving from the first and the second field of the messages. From $\{\overset{K_m}{\mapsto} M, T_m\}_{K_{ca}^{-1}}$, the acquirer A believes that K_m is the merchant M's public key (Assumptions A2, A5, and A8). From $\{M, U, OI_m, n_a, n_m\}_{K_m^{-1}}$ and the freshness of n_a (Assumption A4), the acquirer achieves the belief A **believes** M **believes** (M, U, OI_m). This means that the acquirer A believes that the merchant M has recently issued a payment request for purchase OI_m made by the customer U. Furthermore, as the acquirer A trusts the merchant M to correctly control the payment requests it issues (Assumption A9), then the acquirer achieves the belief A **believes** (M, U, OI_m), i.e., Property P2.

Let us now consider the third and fourth component of message M3. From the first field of certificate $\{\overset{K_d}{\mapsto} D, D$ **controls** U **believes** $X, T_d\}_{K_{ca}^{-1}}$, the acquirer A achieves the belief A **believes** $\overset{K_d}{\mapsto} D$, i.e., K_d is the public key of the personal trusted device D (Assumptions A2, A5, and A8). Furthermore, from the second field of the certificate (Assumptions A2 and A5), the acquirer also achieves the belief A **believes** (CA **believes** (D **controls** U **believes** X)). This means that the acquirer A believes that the certification authority CA has recently stated that the personal trusted device D correctly relays the customer U's beliefs.

As the acquirer trusts the certification authority on this kind of statements (Assumption A14), then the acquirer achieves the following belief about trust on the customer's personal trusted device A **believes** (D **controls** U **believes** X). This means that A believes that the personal trusted device D correctly relays the customer U's beliefs.

From $\{U$ **believes** $(U, M, OI_u), n_a, n_d\}_{K_d^{-1}}$ and the freshness of n_a (Assumption A4), the acquirer achieves the belief A **believes** D **believes** U **believes** (U, M, OI_u). This means that the acquirer A believes that the personal device D has recently relayed a payment order issued by the customer U in favour of merchant M for a purchase OI_u. This belief, together with the one about trust on the customer's personal trusted device, takes the acquirer to believe A **believes** U **believes** (U, M, OI_u). This means that the acquirer believes that the customer has recently uttered a payment request for purchase OI_u from the merchant M. As the acquirer A trusts the customer U to control the purchases for which he utters payment requests (Assumption A10), then the acquirer achieves the belief A **believes** (U, M, OI_u), i.e., Property P1.

Message M4 conveys the response of the acquirer A to merchant M. Then, the merchant M forwards it to the personal trusted device D by means of message M5. The reasoning that merchant and device apply upon receiving message M4 and M5, respectively, are substantially the same. They both believe that the message has been signed by means of the acquirer A's private key (Assumptions A2, A1, A5, and A8). Furthermore, they both believe the message is fresh because it contains the fresh quantities n_m and n_d (Assumption A4), respectively. It follows that they achieve the respective beliefs M **believes** A **believes** (R, A, U, M, OI_m), and D **believes** A **believes** (R, A, U, M, OI_u), i.e., the merchant and the personal trusted device believe that the acquirer has recently issued the response message.

As the merchant trusts the acquirer to correctly generate a response (Assumption A11), then, it achieves the belief M **believes** (R, A, U, M, OI_m), i.e., Property P3.

The personal trusted device D forwards its own belief to the customer U through a secure and timely channel (Assumption A6) with message M6. As the customer trusts the personal trusted device to correctly relay beliefs of the acquirer (Assumption A13), and trusts the acquirer to correctly generate a response (Assumption A11), then the customer achieves the belief U **believes** (R, A, U, M, OI_u), i.e., Property P4.

5.3. On the BAN logic and assumptions

In this work we have used (an extended version of) the BAN logic as a guidance tool. The BAN logic has been quite successful from a practical viewpoint, mostly as a tool for retrospective analysis [8]. However, its semantics are controversial. Completeness for original BAN logic cannot be expected and its extensions have so far been limited to soundness [11]. A version of BAN that is complete with respect to message passing systems has recently been proposed by Cohen and Dam [11]. Furthermore, transforming a protocol into its idealized version and vice versa is quite an informal process. Other logical frameworks could have been used. However, despite its shortcomings, we have used the original BAN logic (with extensions for secure channels) because it allows us to simply and intuitively describe protocols, enclosing properties and assumptions, and clarify trust relationships between parties.

The main objective of this paper is to show that the security of remote card transactions can be improved by means of a personal trusted device, even one with a limited form factor. Given a set of desired security properties, we have defined both a protocol and set of assumptions, A1–A14, from which the protocol achieves the properties. Assumptions A1–A14 have to be considered as a set of practically reasonable hypotheses from which the proposed protocol achieves all the desired properties without generating conflicting beliefs (Section 5.2.1). From this point of view the assumptions can be considered complete and sound. However, we do not claim that the soundness and, particularly, the completeness of these assumptions can be proven in the most general case. First of all, because of the limitations of the BAN semantics discussed above. Second, because this goes beyond our objectives.

Notwithstanding, we believe that our assumptions about personal trusted devices (A1, A3, A6–A7, and A12–A14) may have quite a general value. Actually, they are able to formalize the high-level, informal properties that Pfitzmann *et al.* claim a personal trusted device should have [40]. Furthermore, our assumptions can be implemented on top of the trusted open platform proposed by Lampson *et al.* [14]. More precisely, as to the properties proposed by Pfitzmann et al., *the personal-agent trust* property is captured by Assumptions A6–A7, A12–A14, whereas the *captured-agent trust* property is captured by Assumptions A1 and A3. As to the architecture proposed by Lampson et al., *authenticated operation* makes it possible to implement Assumptions A13; *sealed storage* makes it possible to implement the assumptions A1 and A3; *secure I/O* makes it possible to implement Assumptions A6–A7; and, finally, *attestation* makes A12 and A14 unnecessary because a participant can remotely attestate the personal trusted device and does not need the Certification Authority to attestate the device trustworthiness anymore.

6. Discussion

In this section we exploit the theoretical framework defined in Section 5 to discuss the security of a conventional system, with and without smart cards, and to argue that a GSM/UMTS cellphone can be practically considered a personal trusted device. We essentially focus on the security aspects on the personal trusted device side. In contrast we do not consider the security aspects on the server side because, although important, they are quite well understood and documented [29]. Finally, we briefly discuss an early prototype we have implemented.

6.1. Comparison with a traditional system

In this section we discuss the security of a *conventional* electronic payment system and compare it against the proposed one. A conventional system is a specialization of the system in Fig. 2 in which a customer lacks a personal trusted device and performs the payment phase directly through the host node H. This node runs an electronic payment application, performs the necessary digital signatures on the payment order, and stores the related private key.

In a conventional system it is not generally possible to satisfy Requirement 1 because the acquirer fails to receive a strong proof that the customer has actually authorized the payment. From a more formal point of view, this means that Property P1 does not hold. There are two reasons for that. One reason is that the acquirer A cannot achieve the belief that host H actually speaks for U. More formally, this means that Assumptions A12 and A14 cannot be rephrased by replacing the personal trusted device D with the user host H. This is because it is not generally possible to have assurance about the integrity of the software stack on host H. The computers in most people's homes and offices run operating systems with million lines of code that are known to be full of bugs and security flaws. On top of these operating systems, users run applications with security problems that can be remotely exploited to install malicious software [41]. Furthermore, people are generally unable to properly administer their own computers, are not aware of the security threats, and are used to freely download and install software from the Internet. It follows that an attacker can easily violate the integrity of the software stack of these platforms. When this happens, there is no limit to the damage that a malicious software can do. For instance, the electronic payment application on host H might be maliciously modified by a virus or a Trojan horse so that the order instructions that are being signed are different from those displayed to the customer. As a further example, the maliciously modified application might even perform payments in the name of the user but without his knowledge or consent.

Furthermore, a malicious program, such as a virus or a Trojan horse, could install itself at the operating system level and intercept communication between the payment application and input/output devices so replaying or altering the integrity of messages on I_h and O_h channels. It follows that not even Assumptions A6 and A7 about security of channels hold anymore [43].

A further reason why Property P1 does not hold anymore is that the Acquirer A has no assurance that the public key K_h is a good key for speaking with host H. This means that there is no assurance about the secrecy of the corresponding private key K_h^{-1} and thus the belief $(A \textbf{ believes } \overset{K_h}{\mapsto} H)$ cannot be reliably achieved from Assumptions A1 and A8. Actually, such a confidentiality can be attacked in several ways. Typically, a private key is stored on the host file system in its encrypted form and the symmetric key used to encrypt it is derived from a password or passphrase of the customer's choice. It follows that K_h^{-1} is as secure as that password/passphrase. A malicious program running on the host H (e.g., a virus or a Trojan horse) may attempt an on-line dictionary attack or send the encrypted private key to a remote site for an off-line attack. Furthermore, the malicious program might attempt to directly eavesdrop on the password/passphrase or the private key itself when either is used [43].

As a final consideration, we would like to remark that the customer is exposed to the risk of his private key, as well as his electronic payment application, being compromised and abused without the means of detecting it until the fraudulent use becomes evident. A sophisticated attack might leave no evidence and the customer would remain in the weak position to resist an assertion from the bank that the remote card transaction was correctly authorized.

6.2. Extending a conventional system with a smart card

In order to improve the security of a conventional system, it has been proposed to use a smart card S as a personal trusted device. Every user holds a smart card which is entitled to store the private key K_s^{-1} and to perform digital signatures. The host node H is now equipped with a smart card reader. The electronic payment application on the host node displays the payment order to the customer, and sends it to the smart card in order to have it signed.

Using a smart card certainly improves the secrecy of the signing key with respect to a conventional system. Actually, the key never leaves the card and thus eavesdropping on it becomes impossible or, at least, much more difficult. Therefore, the acquirer can now reliably achieve the belief $(A \textbf{ believes } \overset{K_s}{\mapsto} S)$ from Assumption A1 and A8 properly rephrased in terms of S instead of D.

However, using a smart card cannot make the acquirer believe that the smart card itself actually speaks for the customer. As a smart card has no input and output devices, it has to resort to the host in order to dialogue with the customer. It follows that the implementation of its input and output channels, I_S and O_S, involves the untrusted host H. Thus these channels can be considered neither secure nor timely. Thus, A6 and A7 cannot be assumed. As a consequence, the smart card S cannot infer that the belief it receives from the input channel I_S actually comes from

the customer and thus cannot relay that belief. As a practical example, a compromised host can still send the smart card a payment order that is different from that one displayed to the customer, or perform payments in the name of the user but without his knowledge or consent. The fact that a digital signature by the smart card has to be authorised by entering a PIN is of little help because a compromised host can compromise that PIN.

6.3. A SIM-based payment system

With reference to Fig. 2, we consider now an electronic payment system where the personal trusted device D is a GSM cellphone. We chose this kind of device because GSM is currently one of the most popular and widely used wireless technologies, and GSM handsets are popular, widespread, and considered as the major devices for mobile commerce.

In the rest of this section we consider the GSM application platform that allows an application residing on a GSM cellphone to interact with a service located in the Internet. We give intuitions regarding the technical obligations to develop such an application. Finally, we informally argue that a GSM application platform satisfies the assumptions concerning a personal trusted device. In doing that, we make reference to the standards issued by ETSI, the *European Telecommunications Standards Institute* (www.etsi.org). It is worthwhile to notice that UMTS is the next generation system with respect to GSM and provides a communication system that has higher capacity and is more secure. However, from the application perspective, the arguments for GSM hold for UMTS as well.

In GSM, a cellphone is called *mobile station* and is composed of two parts: the Mobile Equipment and the Subscriber Identity Module. The *Mobile Equipment* (ME) is the handset and is responsible of the radio part. The *Subscriber Identity Module* (SIM) is a security device, a smart card, which contains all the necessary information about subscription, network specific authentication algorithm, and subscriber specific authentication key. The SIM is thus an integral part of the overall network security system which strives against illegitimate use of the service and the interception of data on the air interface [16]. Encryption algorithms are integrated into the mobile equipment as dedicated hardware. Symmetric keys are derived from subscription information using an algorithm under the control of a master key.

In order to use a GSM cellphone in the electronic payment system described in this paper, the SIM is required to contain a small payment application based on the SIM Application Toolkit. The *SIM Application Toolkit* (SAT) is a set of commands and procedures that allow operators and other providers to create applications that reside on the SIM [19]. SAT provides mechanisms by means of which applications can interact and operate with any compliant mobile equipment. These mechanisms include displaying text from the SIM to the mobile equipment, sending and receiving SMS messages, and initiating a dialogue with the user.

An application on the SIM requires end-to-end security services that are not provided by the basic GSM network security services that are instead oriented to identification of subscribers and protection of the air interface. Thus a set of security services has been defined for SAT related to communication over the GSM network (e.g., SMS, USSD, and future transport mechanisms). These services allow a level of security chosen by the network operator or the application provider and include authentication, message integrity, replay detection, proof of receipt, and message confidentiality [18]. An implementation of these services has been proposed over SMS [17].

The above security services require digital signatures as a basic security mechanism [18]. As no public key algorithm is integrated in the mobile equipment, we assume that the SIM implements them under the form of a Wireless Identification Module. The *Wireless Identification Module* (WIM) is a security module conceived to enhance the security of the transport and application layer of the WAP architecture [51]. Application level security operations that use the WIM include signing and using a private key (i.e., K_d^{-1}) that never leaves the WIM. The WIM functionality is designed so that it can be implemented on a smart card with current technology either as a WIM-only card or as a part of a multi-application card containing other card applications, like the GSM SIM. SIM cards with WIM security module are provided by the SIM card issuer [38]. As the private key never leaves the SIM/WIM security module, it follows that it is possible to reliably achieve the belief (A **believes** $\overset{K_d}{\mapsto} D$) from Assumption A1 and A8.

An API has been devised to allow easy application access to the functions and data of the SIM [20] and the SAT [19], so that SIM based services can be developed and loaded onto SIMs, quickly and, if necessary, remotely, after the card has been issued [21]. The API has been proposed in the Java programming language [22] as an extension to the Java Card 2.1 API [47] on the Java Card 2.1 Runtime Environment [48].

According to [21], only the card issuer, or any party delegated by it (i.e., network operator or trusted third parties), can load, remotely or not, an application onto a SIM card. This operation is augmented by certification, mutual authentication and encryption. The role of certification is to ensure that only the authorized entities are able to download an application onto the SIM. Based on this certificate, the card shall decide whether or not to accept the downloaded application. Furthermore, loading an application from a load server (e.g., arranged by the network operator) onto the SIM may involve the authentication of the communicating entities as well as the encryption of the data traffic between those entities [17,18]. To this purpose, the card issuer initializes a set of keys during SIM card initialization and uses them to bootstrap a hierarchy of keys. Additional keys are generated, distributed using existing

keys, and assigned limited authority. Such keys may be passed on to trusted parties and subsequently used for authentication and encryption.

From these considerations it follows that GSM defines a specialized *closed* application platform. Developers (card issuer, network operator, or trusted third parties) have complete control over the structure and complexity of the software stack, thus they can tailor it to their security requirements. In addition, the platform provides hardware tamper resistance to ensure that the platform's software stack is not easily modified to make it misbehave. Furthermore, embedded cryptographic keys permit the GSM system to identify its own software to remote systems, allowing them to make assumptions about the software behavior. It follows that these capabilities allow the GSM closed platform to offer higher assurance and address a wider range of threat models than current general-purpose platforms (e.g., a PC). This leads us to safely assume that now the personal trusted device satisfies Assumptions A12 and A14. Furthermore, as SAT allows the SIM to fully control the user input and output [23], then we can safely assume that also Assumptions A6 and A7 now hold.

6.4. On phishing

Phishing is a criminal activity exploiting social engineering techniques. Phishers attempt to fraudolently acquire sensitive information, such as passwords and credit card details, by masquerading as a trustworthy entity in an electronic communication. Typically, phishing is carried out using email and instant messaging, although phone contact has been used as well.

In a traditional remote card transaction scheme, credit card details are a target for phishers which can use these details to fraudolently carry out remote card transactions. The risk connected to this attack is ultimately shifted to merchants. Actually, if a cardholder repudiates a transaction, the bank has no basis on which to charge the cardholder. Therefore the bank obtains reimbursement from the merchant of anything paid to the merchant in respect of the transaction. It follows that, the subject who takes the risk of phishing is different from the phishing target.

With respect to the traditional scheme for remote card transactions, the proposed one introduces an improvement in terms of anti-phishing measures. Actually, in the proposed scheme, the sole knowledge of the credit card details is not sufficient to authorize a transaction. Actually, a transaction authorization requires a digital signature performed by means of the cardholder's trusted device. This implies that in the proposed scheme the target of a phisher becomes the cardholder's trusted device's private key K_d^{-1}. However, a phisher cannot acquire that key. First, because the key never leaves the trusted device. Second, because it is generally unknown to the customer himself.

Finally, it is worthwhile to notice that the non-repudiation PIN *NRP* is not an attractive target for phishers. This PIN is necessary to authenticate the customer to his personal trusted device. Therefore, there is no point for a customer to type it into an input channel of a device different from his own personal trusted device. Furthermore, even though a phisher were able to acquire the *NRP* of a customer, this information would be useless without the possession of also the customer's personal trusted device.

The last consideration allows us to make a final remark. Like SET, the proposed scheme uses digital signatures. However, the proposed scheme defines a more reasonable risk allocation than SET. For the customer, losing control of the means of authorizing a transaction, namely losing control of his personal trusted device, becomes now similar to losing control of a cellphone or a plastic credit card. An event that the customer can promptly realize and report.

6.5. A prototype

We have developed an early prototype of the payment application for a GSM cellphone. The application on the cellphone was developed as a SAT *applet* in the Java card 2.1 programming language. The applet uses the packages sim.access and sim.toolkit to access the SIM and the SAT services, respectively. The source files for these packages are attached to the standard GSM 03.19 [22].

The SAT applet was not deployed on a real SIM card because of the high economical costs of the simulation and development kits for this kind of smart cards. In contrast, we deployed the applet on a Cyberflex Access 32K [3] using the Cyberflex Access Software Development Kit Version 4.1 [4]. Cyberflex Access 32K is a card equipped with an 8-bit CMOS micro-controller, 96 kB ROM, 4 kB RAM, and 32 kB EEPROM. The cryptographic capabilities of the card are DES, 3-DES (CBC, EBC), SHA-1 hashing, and RSA signature and verification with up to 1024 bits key.

Using a Java Card instead of a real SIM card required us to simulate the SIM–ME interface and to modify the sim.toolkit package accordingly. Furthermore, communication between the cellphone and the acquirer was implemented over UDP instead of SMS. However, the accurate simulation of the SIM–ME interface allowed us to gain experience with the SAT style of programming. This style is peculiar because it reverses the usual modality of interaction between the SIM and the ME. Generally, ME is the "master" and initiates commands to the SIM. However, this limits the possibility of introducing new SIM features requiring the support of the ME, as the ME needs to know in advance what actions it should take. For this reason, the SIM Application Toolkit provides *proactive commands* which allow applications on the SIM to interact and operate with any ME by initiating the actions to be taken by the ME. More detailed information about programming with proactive commands can be found in both the ETSI GSM 11.14 standard [19] and the sim.toolkit documentation that is distributed in the Javadoc format with the package itself.

7. Conclusions

In this paper we have presented a protocol for remote card transactions with personal trusted devices. The paper has several merits.

- The paper shows that personal trusted devices allow us to improve the technical management of disputes so giving rise to a fairer allocation of risks between customer and merchant.
- The paper shows a practical balance between security and usability: conventional, resource-rich but untrusted computer platforms are used for goods browsing and selection while personal, resource-scarce but trusted devices are used to securely authorize card transactions.
- By means of an extended version of the BAN logic, we have stated the security requirements of a personal trusted device. To the best of our knowledge, this is the first effort to make such a formalization.
- This formalization has provided us the theoretical framework to discuss the security of a conventional electronic payment system, with and without smart cards, and to argue on the basis of the ETSI standards that a GSM/UMTS cellphone can be practically considered a personal trusted device as long as it is part of the specialized "closed" GSM application platform.

As we have seen, the security benefits of starting from scratch on a closed special-purpose platform can be significant. However, for most applications these benefits do not outweigh the advantages of general-purpose open platforms that offer significant economies of scale. For this reason, personal wireless devices, enclosing cellphones, are more and more evolving toward being general-purpose open platforms. Due to advances in computer and communication technology, these devices tend to be increasingly powerful in computing and communication. This phenomenon drives more and more functionalities and applications into these devices but it also drives the security risks that are endemic to desktop computing. Unfortunately, manufacturers have largely ignored the lessons from the past and have failed to include features for secure computing at application, operating system and architecture level [30]. However, the recent appearing of trusted platforms seems to provide a promising means to resolve the conflict between these two approaches by supporting the capabilities of closed platforms on general-purpose computing hardware through a combination of hardware and operating system mechanisms [14,28,39]. Our future research activity will be thus addressed to that direction.

Acknowledgements

We wish to thank Professor Gigliola Vaglini from the University of Pisa for helping us to clarify the BAN logic properties of completeness and soundness. The authors are grateful to the anonymous reviewers for their constructive comments.

Appendix A. The extended BAN logic

The statements of the extended BAN logic are reported in Table 2. These statements can be manipulated by using a set of postulates, which include:

The message-meaning rule. This postulate states that if Q sees a message signed with K^{-1} and the cognate key K is a good key for communicating with P, then Q believes that once P said the message. Formally,

$$\frac{Q \textbf{ sees } \{X\}_{K^{-1}}, \overset{K}{\mapsto} P}{P \textbf{ said } X}$$

The postulate can be restated for shared secrets as follows: if Q sees a message X combined with Y, which is a secret shared with P, then Q believes that once P said the message. Formally,

$$\frac{Q \textbf{ sees } \langle X \rangle_Y, Q \textbf{ believes } P \overset{Y}{\rightleftharpoons} Q}{P \textbf{ said } X}$$

Table 2
The BAN formalism

Formalism	Description
P **believes** X	Principal P is entitled to believe statement X and to behave as if X were true
P **said** X	Principal P once said statement X, either long ago or in the current execution instance of the protocol
P **said**$_L$ X	Principal P once said statement X on channel L
P **sees** X	A message containing X has been sent to principal P that can read and repeat X
P **sees**$_L$ X	A message containing X has been sent to principal P on channel L
fresh (X)	Statement X is fresh, that is, it has been uttered during the current execution instance of the protocol
P **controls** X	Principal P is an authority on statement X and thus must be trusted on it
$\overset{K}{\mapsto} P$	Key K is a good key to communicate with P; i.e., K is the public key of principal P
$P \overset{X}{\rightleftharpoons} Q$	X is a shared secret between principals P and Q
$\langle Z \rangle_X$	Z is combined with secret X whose presence proves the identity of whoever utters $\langle Z \rangle_X$. Combination can be as simple as a hash or a concatenation
$\overset{Y}{\prec} P$	Y is a secure channel from principal P
timely (Y)	Y is a timely channel

The left column shows the BAN formalism we use in this paper, and the right column reports an informal description of such a formalism.

Abadi et al. have restated the statement for secure channels as follows: if Q sees a message X on channel L, which is a secure channel from P, then Q believes that once P said X [1]. Formally,

$$\frac{Q \textbf{ sees}_L\ X, Q \textbf{ believes } P \overset{L}{\prec}}{P \textbf{ said } X}$$

The nonce-verification rule. This postulate states that if P once uttered statement X, and this statement is fresh, then P still believes it. Formally,

$$\frac{Q \textbf{ believes } P \textbf{ said } X, Q \textbf{ believes fresh } (X)}{Q \textbf{ believes } P \textbf{ believes } X}$$

Another way to guarantee timeliness is to use timely communication links [1]. That is, if once P uttered a message X on channel L, and this channel is timely, then P still believes X. Formally,

$$\frac{Q \textbf{ believes } P \textbf{ said}_L\ X, Q \textbf{ believes timely } (L)}{Q \textbf{ believes } P \textbf{ believes } X}$$

The jurisdiction rule. This postulate states that, if principal P believes a statement X, and P is an authority on that matter, then he should be believed. Formally,

$$\frac{Q \textbf{ believes } P \textbf{ believes } X, Q \textbf{ believes } P \textbf{ controls } X}{Q \textbf{ believes } X}.$$

References

[1] Martin Abadi, Michael Burrows, Charles Kaufman, Butler Lampson, Authentication and delegation with smart-cards, Science of Computer Programming 21 (1993) 93–113.

[2] N. Asokan, Philippe A. Janson, Michael Steiner, Michael Waidner, The state of the art in electronic payment systems, IEEE Computer (1997) 28–35.

[3] Axalto. Cyberflex Access. Available from: <http://www.cyberflex.com/Products/cards/access.html>.

[4] Axalto. Cyberflex Access SDK. Available from: <http://www.cyberflex.com/Products/sdk.html>.

[5] Feng Bao, L. Anantharaman, R. Deng, Design of portable mobile devices based e-payment system and e-ticketing system with digital signature, in: Proceedings of the International Conference on Info-tech and Info-net (ICII 2001), vol. 6, Beijing, China, October 29–November 1 2001, pp. 7–12.

[6] Mihir Bellare, Juan Garay, Ralf Hauser, Amir Herzberg, Hugo Krawczyk, Michael Steiner, Gene Tsudik, Els Van Herreweghen, Michael Waidner, Design, implementation and deployment of the *i*KP secure electronic payment system, IEEE Journal on Selected Areas in Communications 18 (4) (2000) 611–627.

[7] Nicholas Bohm, Ian Brown, Brian Gladman, Electronic commerce: Who carries the risk of fraud? The Journal of Information and Law Technology (2000) 31–34.

[8] Mike Bond, Jolyon Clulow, Extending security protocol analysis: New challenges, in: Proceedings of the First Workshop on Automated Reasoning for Security Analysis (ARSPA), Cork, Ireland, July 4 2004, pp. 13–24.

[9] Michael Burrows, Martin Abadi, Roger Needham, A logic of authentication, ACM Transactions on Computer Systems 8 (1) (1990) 18–36.

[10] Joris Claessens, Bart Preenel, Joos Vandewalle, Combining world wide web and wireless security, in: Bart De Decker, Frank Piessens, Jan Smits, Els Van Herreweghen (Eds.), Proceedings of IFIP I-NetSec 2001, Advances of Network and Distributed Systems Security, International Federation for Information Processing, Kluwer Academic Publishers, Leuven, Belgium, 2001, pp. 153–171.

[11] Mika Cohen, Mads Dam, A completeness result for ban logic, in: Proceedings of the 4th International Workshop on Methods for Modalities (M4M-4), Berlin – Adlershof, Germany, December 1–2 2005.

[12] Yuanjun Dai, Lihe Zhang. A security payment scheme of mobile e-commerce, in: Proceedings of the International Conference on Communication Technology (ICCT 2003), Beijing, China, April 9–11 2003, pp. 949–952.

[13] T. Dierks, C. Allen. The TLS protocol version 1.0. The Internet Engineering Task Force (IETF), RFC 2246, January 1999.

[14] Paul England, Butler Lampson, John Manferdelli, Marcus Peinado, Bryan Willman, A trusted open platform, IEEE Computer 36 (7) (2003) 55–63.

[15] European Telecommunications Standards Institute. Digital cellular telecommunication system (Phase 2+); Technical realization of the Short Message Service (SMS) Point-to-Point (PP). ETSI TS 100 559 (3GPP TS 04.11).

[16] European Telecommunications Standards Institute. Digital cellular telecommunications system (Phase 2); Security aspects. ETS 300 506 (GSM 02.09).

[17] European Telecommunications Standards Institute. Digital cellular telecommunications system (Phase 2+); Security Mechanisms for the SIM application toolkit; Stage 2. ETSI TS 101 181 (GSM 03.48).

[18] European Telecommunications Standards Institute. Digital cellular telecommunications system (Phase 2+); Security mechanisms for the SIM Application Toolkit;Stage 1. ETSI TS 101 180 (GSM 02.48).

[19] European Telecommunications Standards Institute. Digital cellular telecommunications system (Phase 2+); Specification of the SIM Application Toolkit for the Subscriber Identity Module–Mobile Equipment (SIM–ME) interface. ETSI TS 101 267 (GSM 11.14).

[20] European Telecommunications Standards Institute. Digital cellular telecommunications system (Phase 2+); Specification of the Subscriber Identity Module - Mobile Equipment (SIM–ME) interface. ETSI TS 100 977 (3GPP TS 11.11).

[21] European Telecommunications Standards Institute. Digital cellular telecommunications system (Phase 2+); Subscriber Identity Module Application Programming Interface (SIM API); Service description; Stage 1. ETSI TS 101 413 (GSM 02.19).

[22] European Telecommunications Standards Institute. Digital cellular telecommunications system (Phase 2+); Subscriber Identity Module Application Programming Interface (SIM API); SIM API for Java Card; Stage 2. ETSI TS 101 476 (GSM 03.19).

[23] European Telecommunications Standards Institute. Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); USIM/SIM Application Toolkit (USAT/SAT); Service description; Stage 1. ETSI TS 122 038 (3GPP TS 22.038).

[24] Wireless Application Protocol Forum. Wireless Datagram Protocol Specification, 14 June 2001. Available from: <www.wapforum.org>.

[25] Wireless Application Protocol Forum. Wireless profiled TCP Specification, 31 March 2001. Available from: <www.wapforum.org>.

[26] Alia Fourati, Hella Kaffel Ben Ayed, Farouk Kamoun, A bdelmalek Benzekri. A set approach to secure the payment in mobile commerce, in: Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN02), Tampa, Florida (USA), 6–8 November 2002, pp. 136–137.

[27] A. Freier, P. Kariton, P. Kocher, The SSL protocol: Version 3.0. Netscape Communication, Inc., Mountain View, CA, March 1996. Available from: <http://home.netscape.com/eng/ssl3/ssl-toc.html>.

[28] Tal Garfinkel, Ben Pfaff, Jim Chow, Mendel Rosenblum, Dan Boneh, Terra: a virtual machine-based platform for trusted computing, in: Proceedings of the Nineteenth ACM Symposium on Operating

Systems Principles, Bolton Landing, NY, USA, October 19–22 2003, pp. 193–206.
[29] Anup K. Ghosh, Security and Privacy for E-Business, Wiley, New York (USA), 2001.
[30] Anup K. Ghosh, Tara M. Swaminatha, Software security and privacy risks in mobile e-commerce, Communications of the ACM 44 (2) (2001) 51–57.
[31] Jim Gray, Andreas Reuter, Transaction Processing: Concepts and Techniques (Morgan Kaufmann Series in Data Management Systems), first ed., Morgan Kaufmann, 1993.
[32] Theo Härder, Andreas Reuter, Principles of transaction-oriented database recovery, ACM Computing Survey 15 (4) (1983) 287–317.
[33] Amir Herzberg, Payments and banking with mobile personal devices, Communications of the ACM 46 (5) (2003) 53–58.
[34] Liang Jin, Liang Feng, Gao Zheng Hua, Research on wap clients supports set payment protocol, IEEE Wireless Communications 9 (1) (2002) 90–95.
[35] Vorapranee Khu-smith, Chris J. Mitchell, Using gsm to enhance e-commerce security, in: Proceedings of the Second ACM International Workshop on Mobile Commerce, Atlanta, Georgia, USA, September 28–28 2002, pp. 75–81.
[36] Won Kim, Highly available systems for database applications, ACM Computing Survey 16 (1) (1984) 71–98.
[37] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.
[38] Oberthur Card System. SIMphonIC™. Available from: <http://www.oberthurusa.com>.
[39] Siani Pearson, Trusted Computing Platforms—TCPA Technology in Context, Hewlett-Packard Company, Prentice Hall PTR, 2003.
[40] Andreas Pfitzmann, Birgit Pfitzmann, Matthias Schunter, Michael Waidner, Trustworthy user devices, in: Günter Müller, Kai Rannenberg (Eds.), Multilateral Security in Communications, Addison-Wesley, 1999, pp. 137–156.
[41] Aviel D. Rubin, Security considerations for remote electronic voting, Communications of the ACM 45 (12) (2002) 39–44.
[42] Bruce Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, second ed., Wiley, 1996.
[43] Bruce Schneier. Why digital signatures are not signatures. In Crypto-Gram Newsletter. November 2000. Available from: <http://www.schneier.com/crypto-gram-0011.html#1>.
[44] S. Schwiderski-Grosche, H. Knospe, Secure mobile commerce, Electronics & Communication Engineering Journal 14 (5) (2002) 228–238.
[45] Secure Electronic Transaction LLC. Secure Electronic Transaction Specification. Available from: <http://www.setco.org/set_specifications.html>.
[46] James A. Senn, The emergence of m-commerce, IEEE Computer 33 (12) (2000) 148–150.
[47] SUN. Java Card 2.1 API Specification. Java card Specification.
[48] SUN. Java Card 2.1 Runtime Environment Specification. Java card Specification.
[49] Do Van Thanh, Security issues in mobile ecommerce, in: Proceedings of the 11th IEEE International Workshop on Database and Expert Systems Applications, London, UK, August 4–8 2000, pp. 412–425.
[50] Upkar Varshney, Mobile payments, IEEE Computer 35 (12) (2002) 120–121.
[51] WAP Forum. Wireless Identity Module. Part: Security. WAP-260-WIM-20010712-a.
[52] WAP Forum. Wireless application protocol – architecture specification. WAP-210-WAPArch-20010712, July 2001.
[53] Wireless Application Protocol Forum. WAP Transport Layer End-to-end Security, 2001. WAP-187-TransportE2ESec-20010628-a.
[54] Konrad Wrona, Marko Schuba, Guido Zavagli, Mobile payments – state of the art and open problems, in: L. Fiege, G. Mühl, U. Willhelm (Eds.), WELCOM 2001, Lecture Notes in Computer Science, LNCS, 2232, Springer, Berlin Heidelberg, 2001, pp. 88–100.

Andrea Bottoni was born in Pisa, Italy, in 1977. He received his Master's degree in computer engineering in 2002, and his Doctoral degree in computer security in 2006, both from the University of Pisa, Italy. His research interests include the design and analysis of security infrastructures in multi-domain distributed systems. He is currently information security consultant for EDS Italy, Rome.

Gianluca Dini received his "Laurea" degree in Electronics Engineering from the University of Pisa, Pisa, Italy, in 1990, and his PhD in Computer Engineering from Scuola Superiore "S. Anna", Pisa, Italy, in 1995. From 1993 to 1994 he was Research Fellow at the Department of Computer Science of the University of Twente, The Netherlands. From 1993 to 1999 he was Assistant Professor at the Dipartimento di Ingegneria della Informazione, Elettronica, Informatica, Telecomunicazioni of the University of Pisa, where he is now Associate Professor. His research interests are in the field of distributed computing systems with particular reference to security and fault-tolerance.