*i*KP – A Family of Secure Electronic Payment Protocols

WORKING DRAFT*

Mihir Bellare†, Juan A. Garay†, Ralf Hauser‡, Amir Herzberg†,
Hugo Krawczyk†, Michael Steiner‡, Gene Tsudik‡, Michael Waidner‡

May 8, 1995

Abstract

This paper proposes a family of protocols – *i*KP – for secure electronic payments over the Internet. The protocols implement credit card-based transactions between the customer and the merchant while using the existing financial network for clearing and authorization. The protocols can be extended to apply to other payment models, such as debit cards and electronic checks. They are based on public-key cryptography and can be implemented in either software or hardware. Individual protocols differ in key management complexity and degree of security. It is intended that their deployment be gradual and incremental.

The *i*KP protocols are presented herein with the intention to serve as a starting point for eventual standards on secure electronic payment.

1 Introduction

Nowadays it is hardly necessary to stress the importance of electronic commerce. Suffice it to say that it is rapidly gaining momentum and is equally appealing to both (electronic) merchants and consumers. One aspect lagging behind is the availability of *secure* electronic payment methods. At the same time, it is becoming rather obvious that the appeal of electronic commerce without electronic payment is limited. Moreover, *insecure* electronic payment methods are most likely to impede, rather than promote, electronic commerce.

In this paper we propose a family of secure electronic payment protocols – *i*KP (*i*-Key-Protocol). The protocols are compatible with the existing business models and payment system infrastructure. This is achieved by including an entity called a *gateway* that translates electronic payment requests (generated by customers and merchants via the protocols) into existing clearing and authorization messages. This paper focuses on the credit card

*The most recent version of this document is available from <http://www.zurich.ibm.com/-Technology/Security/extern/ecommerce/>.

†IBM T.J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598, USA, {mihir, garay, amir, hugo}@watson.ibm.com

‡IBM Zürich Research Laboratory, Sämerstrasse 4, CH-8803 Rüschlikon, Switzerland, {rah, sti, gts, wmi}@zurich.ibm.com

payment model as it is anticipated to be the most popular in the near future. However, the protocols can be extended to apply to other payment models, e.g., debit cards and electronic checks.

*i*KP protocols offer varying levels of security and different degrees of sophistication depending on the key management infrastructure. All protocols are based on public-key cryptography but they vary in the number of parties that possess their own public key-pairs. This number is indicated by the name of the individual protocols: 1KP, 2KP, and 3KP.

The simplest protocol, 1KP, requires that only the gateway possess a pair of public and private keys. Customers and merchants need only to possess the authentic public key of the gateway, or the authentic public key of an "authority" that validates the gateway's public key via a signed certificate. This involves a minimal certification authority infrastructure that provides certificates for a small number of entities, namely, the gateways. Such a minimal certification authority can be run, for example, by the credit card company itself. (In which case, a customer will keep, say, the "VISA public key" or "MasterCard public key", etc.) In the 1KP scenario, customers are authenticated on the basis of their credit card numbers and associated secret PINs. Payments are authenticated by communicating the PIN and credit card number *encrypted* under the acquirer's public key, and properly bound to relevant information (purchase amount, id's, etc.). While 1KP is very simple, it does not offer non-repudiation for messages sent by customers and merchants; this means that disputes about the authenticity of payment orders are not easily resolvable. Some consequences of missing non-repudiation for payment systems are illustrated in [1].

2KP improves on 1KP by providing also merchants, in addition to gateways, with public key-pairs and public key certificates. As a result, non-repudiation is provided for messages originated by merchants. Additionally, 2KP enables customers to verify that they are dealing with *bona fide* merchants by checking their certificates, without any on-line contact with a third party. As in 1KP, payment orders are authenticated via the customer's PIN (encrypted before transmission).

Full multi-party security is provided by 3KP, which achieves non-repudiation for all messages and parties involved. Payment orders are authenticated by both a PIN and a digital signature of the customer. This makes the forging of payment orders computationally infeasible. Additionally, 3KP enables merchants to authenticate customers on-line. Notice that in this case a full certification authority infrastructure is required to provide certificates of the customer's public keys.

All *i*KP protocols can be implemented either in software or hardware. In fact, in 1KP and 2KP the customer does not even need a personalized payment device: only credit card data and the PIN must be entered to complete a payment. However, for the sake of increased security, it is desirable to use a tamper-resistant device that can protect the PIN and – in case of 3KP – the secret key of the customer.

Note on the privacy of customer's orders: The iKP protocols do not explicitly provide encryption of the order information. Such protection is assumed to be provided by other existing mechanisms, e.g., SHTTP [18] or SSL [14]. The decoupling of order encryption from the electronic payment protocol is an important design principle of *i*KP which supports compatibility with different underlying browsing and privacy-protecting mechanisms. It also adds to the simplicity, modularity, and ease of analysis of the protocols. An additional advantage is freeing *i*KP from export restrictions related to the use of bulk

encryption. Nonetheless, if desired, the *i*KP family (especially, 2KP and 3KP) can be easily extended to generate shared keys between customer and merchant for protection of browsing and order information.

Organization of the paper: Section 2 briefly discusses other proposed payment mechanisms over the Internet. Section 3 introduces the payment model, including the participants and basic security threats. Section 4 presents the security requirements for each of the players in the protocol. In Section 5 the *i*KP protocols are described and analyzed as for the requirements they satisfy. Conclusions and comparison of the protocols are presented in Section 6.

2 Related Work

The *i*KP family has many features and motivations in common with other proposals for on-line payment systems[1].

Most proposals for on-line payment are based on standard models (e.g, credit cards) and connect the electronic and the conventional payment system via some sort of gateway (e.g., [21, 9, 12]).

As already mentioned, most current on-line payments are not protected at all. Some systems propose symmetric cryptography for efficiency reasons (e.g., [11, 20, 17]), in particular, those that aim at micro-payments. However, most proposals use public key cryptography in a way similar to one or another *i*KP protocol. For example, [12] uses public-key cryptography between merchant and gateway, and the protocol sketched in [9] appears cryptographically similar to 2KP.

The most cryptographically advanced electronic payment systems emphasize *untraceability* and *anonymity* against the payment system [3, 5, 6, 4, 10, 16].

Finally, there are some general security schemes for the *World Wide Web*, most notably, SHTTP [18] and SSL [14]. Both have been suggested as a basis for secure electronic payments. SHTTP is a possible platform for implementing *i*KP. SSL is more thought to secure the link between WWW client and server, and is therefore less suited for multi-party protocols like *i*KP. Moreover, SSL does not support non-repudiation.

The particular advantages of *i*KP over existing proposals are:

- The *i*KP family allows for a gradual deployment. 1KP is based on what already exists today – credit cards, PINs, and the existing payment system networks – and presents a feasible short-term solution. Introduction of public key cerfitication of merchants will usher in 2KP and, as soon as the certification infrastructure for customers is in place, 3KP can be phased in and achieve full multi-party payment security.
- *i*KP is an evolving design, rather than a fixed, closed protocol. It is intended as a starting point for a standard for secure payments over the Internet. It is open for discussion, and we encourage comments on its qualities and suggestions for improvement.

[1] For a comprehensive listing see [15] or <http://www.zurich.ibm.com/Technology/Security/sirene/outsideworld/ecommerce.html>.

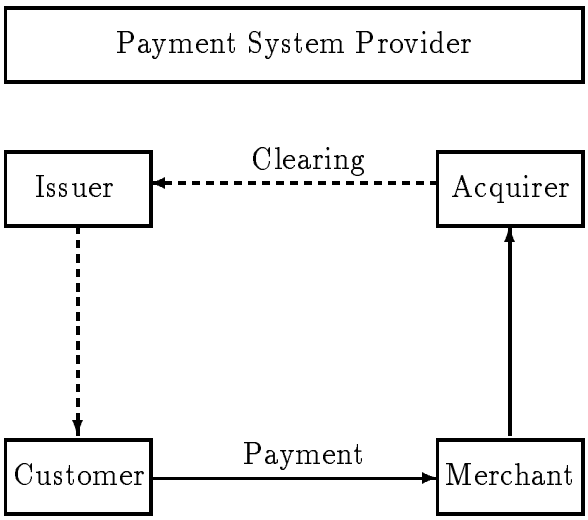

Figure 1: Generic Model of a Payment System

- The use of encryption in iKP is limited to well-defined payment data – credit card numbers and PINs – and the interfaces to cryptographic primitives can be designed in a way that makes them inaccessible to the end-user. Therefore, we expect that iKP (at least specific implementations as sketched in [13]) will not be subject to US export regulations. We note that, for countries outside North America, there is absolutely no incentive to use payment systems with restricted or weakened security.

A specific software-only architecture used for implementing a prototype of iKP is described in [13]. It is independent of any HTTP-extension and works with any WWW browser.

3 Payment Model

3.1 Players

All iKP protocols are based on the existing credit-card payment system. The players of the payment system are shown in Figure 1.

The payment system is operated by a *payment system provider* like Europay, MasterCard, VISA. This payment system provider has fixed business relations with certain banks who act as *issuers* of credit cards to customers or as *acquirers* of payment records from merchants. Each issuer has a Bank Identification Number, BIN, which it receives at the time it signs up with a payment system provider, and which is embossed on each credit card issued as part of the credit card number. The BIN also identifies the payment system provider.

A *customer* receives a credit card from an issuer, and is in possession of a PIN as is common in current systems. In 1KP and 2KP, payments will be authenticated only by means of the credit card number and this PIN (both suitably encrypted!), while in 3KP, a digital signature is additionally used.

It is assumed that (as can be expected for electronic payment) that the customer is using a computer to execute the payment protocol. Since this computer must receive the customer's

PIN or secret signature key, it must be a trustworthy device. We caution that even a customer-owned computer is vulnerable: it may be used by several persons or it may contain a Trojan horse or a virus that could steal PINs and secret keys. The best payment device would be a secure isolated computer, e.g., a tamper-resistant smartcard, connected to the computer used for shopping via a customer-owned smartcard reader with its own keyboard and display. (This is often called an *Electronic Wallet.*) Technically, 1KP and 2KP can be used with any kind of payment device, while for 3KP the customers need personal devices that store their secret signature keys and certificates.

A *merchant* signs up with the payment system provider and with a specific bank, called an *acquirer,* to accept deposits. Like a customer, a merchant needs a secure device that stores the merchant's secret keys and performs the payment protocol.

Clearing between acquirers and issuers is done using the existing financial networks.

The *i*KP protocols deal with the *payment* transaction only (i.e., the solid lines in Figure 1), and therefore involve only three parties, called C – Customer, M – Merchant, and A – Acquirer Gateway. Note that A is no acquirer in the financial sense, but a *gateway* to the existing credit card clearing/authorization network. In other words, the function of A is to serve as a front-end to the *current infrastructure that remains unchanged.*

The protocols presented here describe the core of a payment system only. Besides this, additional mechanisms are needed, e.g., for *cancelation of payment orders* and for *providing statements of account.*

3.2 Public Keys and Certification

Since all *i*KP protocols are based on public-key cryptography, we need a mechanism to authenticate these public keys. We assume a *certification authority,* CA, which has a secret key, SK_{CA}. Its public counterpart, PK_{CA}, is held by all other parties (see Section 5.4 for details.) Basically, CA will certify a public key of party X by signing the pair (X, PK_X) consisting of the identity of X and X's public key. (The signature is computed under SK_{CA}.)

Note that PK_{CA} must be conveyed in an authenticated manner to every party. This will be typically done out-of-band, via any of a number of well-known mechanisms (see Section 5.4).

For simplicity's sake, we assume that there is only one certification authority. However it is easy to extend the protocols to support multiple certification authorities, e.g., such that the payment system provider at the top-level authority issues certificates to its constituent issuers and acquirers, while these, in turn, issue certificates to their customers and merchants.

In all *i*KP protocols, an acquirer A has a secret key, SK_A, which enables signing and decryption. Its public counterpart, PK_A, (which enables signature verification and encryption) is held by each accredited merchant together with its corresponding CA's certificate. As in current operation, acquirers receive the customers' credit card numbers and PINs, and are trusted to keep these values confidential.

In 2KP, each merchant, and in 3KP also each customer, have a secret/public key-pair. They are denoted by (SK_M, PK_M) and (SK_C, PK_C), respectively. Both public keys are included in certificates issued by CA.

3.3 Adversaries and Threats

We consider three different adversaries:

- **eavesdropper** who listens to messages and tries to learn secrets (e.g., credit card numbers, PIN's)
- **active attacker** who introduces forged messages in an attempt to cause the system to misbehave (e.g., to send him goods instead of to the customer)
- **insider** who either is some legitimate party or learns that party's secrets. (One example is a dishonest merchant who tries to get paid without authorization.)

Before listing requirements in Section 4 we briefly discuss common threats and attacks.

The *Internet* is a decentralized, heterogeneous network, without single ownership of the network resources and functions. In particular, one cannot exclude the possibility that messages between the legitimate parties would pass through a maliciously controlled computer. Furthermore, the routing mechanisms in the Internet are not designed to protect against malicious attacks. Therefore, it is folly to assume either confidentiality or authentication for messages sent over the Internet, unless proper cryptographic mechanisms are employed. To summarize, it is easy to steal information off the Internet. Therefore, at least credit card numbers and PINs must *not* be sent in the clear.

In addition, one must be concerned about the trustworthiness of the merchants providing Internet service. The kind of business that is expected in the Internet would include the so-called *cottage industry* – small merchants. It is very easy for an adversary to set up shop and put up a fake electronic *storefront* in order to get customers' credit card numbers. This implies that the credit card number should travel from customer to acquirer without being revealed to the merchant (who needs only the BIN which can be provided separately.)

Obviously, a good deal of care must be taken to protect the keys of acquirers. One of the biggest concerns is that of an adversary breaking into an acquirer computer through the Internet connection. Therefore, the acquirer's computer must be protected with the utmost care; including a very limited Internet connection using advanced firewall technology (e.g., [8, 7].)

Furthermore, the trust in the acquirer's computer must be limited, so that a break in would have a limited effect only.

4 Security Requirements

In this section we consider a range of potential requirements for each party involved in the payment process: issuer/acquirer, merchant, and customer. They range from mandatory security requirements to optional features.

4.1 Issuer/Acquirer Requirements

The issuer and the acquirer are assumed to enjoy some degree of mutual trust. Moreover, an infrastructure enabling secure communication between these parties is already in place. Therefore, we join the requirements of the issuer and the acquirer.

A1 *Proof of Transaction Authorization by Customer.* When the acquirer debits a certain credit card account by a certain amount, the acquirer must be in possession of an unforgeable *proof* that the owner of the credit card has authorized this payment.

Note that the information certified must include at least the amount and currency of the payment, the date and time, and the merchant identification. Note also that in this context the merchant may be an adversary, and even such a merchant must not be able to generate a fake debit.

We distinguish between:

a. *Weak Proof*, which authenticates the customer to the acquirer but does not serve as a proof for third parties, and

b. *Undeniable Proof*, which provides full non-repudiation, i.e., can be used to resolve disputes between the customer and the payment system provider.

The same distinction will be made for all subsequently required proofs of transaction.

A2 *Proof of Transaction Authorization by Merchant.* When the acquirer authorizes a payment to a certain merchant, the acquirer must be in possession of an unforgeable *proof* that this merchant is willing to accept the payment.

4.2 Merchant Requirements

M1 *Proof of Transaction Authorization by Acquirer.* The merchant needs an unforgeable proof that the acquirer has authorized the payment.

This includes certification and authentication of the acquirer, so that the merchant knows he is dealing with the real acquirer, and certification of the actual authorization information. Note that again the amount and currency, the time and date, and information to identify the transaction must be certified.

We also distinguish between [a] *Weak proof* and [b] *undeniable proof*, which provides full non-repudiation.

M2 *Proof of Transaction Authorization by Customer.* Even before the merchant receives the transaction authorization from the acquirer, the merchant might need an unforgeable proof that the customer has authenticated it. Again we distinguish between [a] *Weak Proof* and [b] *Undeniable Proof.*

This requirement is necessary to provide for *off-line authorization.*

4.3 Customer Requirements

C1 *Unauthorized Payment is Impossible.* It must not be possible to charge something to a customer's credit card without possession of the credit card number, PIN, and in case of 3KP, the customer's secret signature key.

Thus, neither Internet rogues nor malicious merchants must be able to generate spurious transactions which end up approved by the acquirer. This must remain the case even if the customer has engaged in many prior legitimate transactions. In other words,

information sent in one (legitimate) transaction must not enable a later spurious transaction. So in particular the PIN must not be sent in the clear, and not even subject to guessing attacks!

Similar to the two type of proofs of transactions, we distinguish between:

a. *Impossibility*, which means that unauthorized payments are impossible provided the acquirer's secret key is not available to the adversary, and

b. *Disputability*, which means that even if the acquirer's secret key is available to the adversary (e.g., because the adversary co-operates with an insider), the customer can prove that he/she did not authorize the payment.

In fact, C1.a and C1.b are consequences of A1.a and A1.b, respectively.

C2 *Proof of Transaction Authorization by Acquirer.* The customer would like to be in possession of proof that the acquirer authorized the transaction. This "receipt" from the acquirer is not of paramount importance, but is convenient to have.

Again, we distinguish between [a] *Weak Proof* and [b] *Undeniable Proof* (full non-repudiation).

C3 *Certification and Authentication of Merchant.* The customer needs a proof that the merchant is accredited at an acquirer (which could be considered as some guarantee for the trustworthiness of the merchant).

C4 *Receipt from Merchant.* The customer wants a proof that the merchant who has made the offer has received payment and promised to deliver the goods. This takes the form of an undeniable receipt.

2KP and 3KP will satisfy this requirement, but will not ensure fairness: The merchant can always refuse sending this receipt while already having received the authorization message from the acquirer gateway. In this case, the customer must take the next statement of account as a replacement for this receipt.

While Requirements C1 – C4 are discussed in the following, Requirement C5 will not be explicitly addressed. Instead, the relationship of *i*KP to C5 is discussed here.

C5 *Privacy and Anonymity.* Customers want privacy of their order and payment information. For example a businessman may be purchasing the latest information on certain stocks and may not want competitors to know which stocks he is interested in.

a. *Privacy.* The privacy of order information and amount of payment should be implemented independently of the payment protocol, e.g., based on SHTTP [18] or SSL [14]. *i*KP does not reveal order information to any other party than the merchant, at least as long as there is no dispute, but does not include encryption of these data.

Obviously, credit card number and PIN must be protected carefully, which is achieved within *i*KP by encrypting them with the acquirer's public key. (This is the only application of *encryption* in *i*KP, which is made in order to facilitate exportability from the US.)

b. *Anonymity.* Besides confidentiality of order and payment information, customers may want *anonymity* from eavesdroppers and (optionally) also from the merchant.

It is also conceivable that the customer may even want anonymity with respect to the payment system provider.

*i*KP supports anonymity from the merchants in the sense that the customer's identity, address, etc., is not revealed to the merchant.

*i*KP does not offer anonymity from the payment system provider. This might be desirable for systems that aim to replace cash but is not essential for protocols, like *i*KP, that follow the credit card-based payment model.

5 The *i*KP Family

The main technical difference between the three *i*KP protocols is the extent to which public key signatures are used. As mentioned earlier, public key encryption and public key-based certification is practically unavoidable in future electronic payment environments. However, the degree of its use can be varied to result in different security guarantees and different levels of impact/burden for all parties involved.

Throughout the remainder of the paper, the following notation is used:

(PK_i, SK_i) Public and secret key of Party i (Certification Authority CA, Customer C, Merchant M, Acquirer Gateway A).

While A's key pair must enable signature and encryption, all other key pairs need to enable signatures only. Note that signing and encryption are *independent* operations; in particular, $E_i(S_i(x)) \neq x$.

$CERT_i$ Public key certificate of Party i, issued by CA (includes PK_i and CA's signature on PK_i).

$\mathcal{H}(\cdot)$ A strong collision-resistant one-way hash function.

$E_i(\cdot)$ Public-key encryption using PK_i (required to provide message integrity check, in addition to confidentiality; see below).

$S_i(\cdot)$ Signing with SK_i and adding the signed information (i.e., $S_i(info)$ stands for the pair {*info*, signature of Party i on *info*}).

Possible implementations of these primitives are described, e.g., in [19]. A prototype implementation done at IBM Research uses RSA with key length 1024 for signature and encryption, and MD5 as a hash function.

We require that public key encryption not only provides confidentiality but also *message integrity:* decryption of a ciphertext results either in a message or in a flag indicating non-validity. Correct decryption convinces the decryptor that the transmitter *knows* the plaintext that was encrypted. In particular, tampering with ciphertext is detectable. Such encryption has been called *plaintext-aware encryption,* and simple schemes to achieve it in simple ways using RSA (or other common public key functions) are described in [2]. We note that the encryption function is *randomized:* E_i invoked upon message m will use, to compute its output, some randomizer, so that each encryption is different from previous ones.

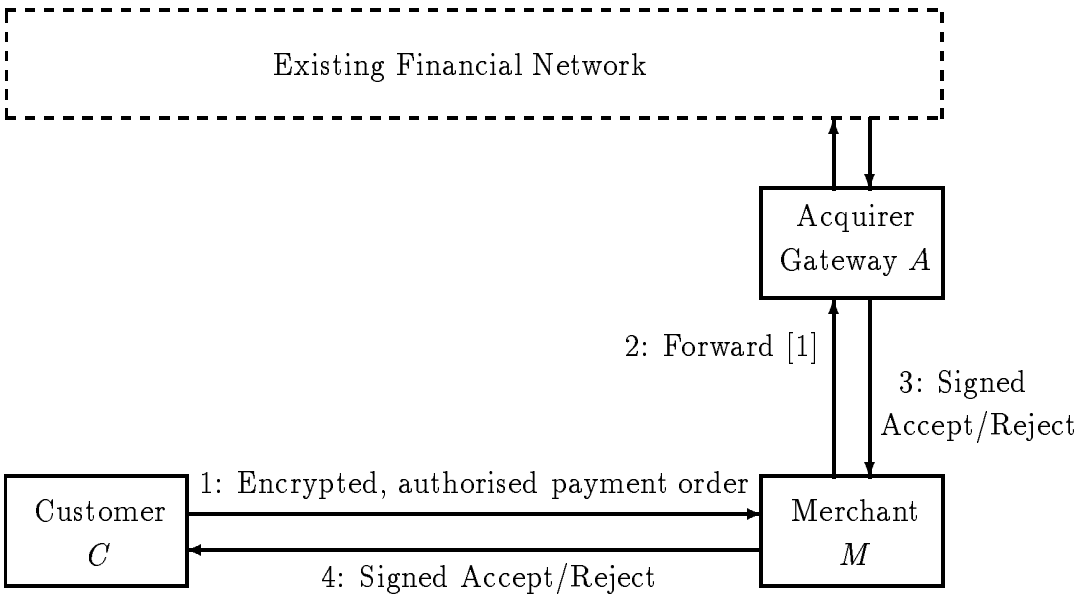

Figure 2: Scenario of *i*KP

We stress that such encryption does not provide authentication in the manner of a signature, i.e., it does not provide non-repudiation. But it can be made to provide an authentication like capability between parties sharing a key (such as the PIN).

All three protocols implement the same scenario, shown in Figure 2.

5.1 1KP

1KP (illustrated in Figure 3) represents the initial step in the gradual introduction of a public-key infrastructure. Although it requires the use of public-key encryption by all parties, only the acquirer gateway, A, needs to possess and distribute its own public key certificate, $CERT_A$. In particular, the total number of certificates to be issued by the certification authority is small as it depends only on the number of gateways.

Like all members of the *i*KP Family, 1KP requires that all customers and merchants have an authentic copy of PK_{CA}, the public key of the certification authority. Every customer C has a secret PIN which is also known to the payment system (but not to the merchants!). Every merchant has to know the certificate of the corresponding acquirer gateway, $CERT_A$.

1KP does not assume A to keep a state per customer. Instead, the customer's PIN is verified using the existing authorization infrastructure (which uses tamper-resistant technology for processing and verification of PIN's).

All parties in 1KP must perform certain public key computations: encryption is only applied *once,* for sending credit card data and PIN from the customer to the acquirer gateway, securely. Therefore, public key encryption is required from C only, while decryption is required from A only (this is true also for 2KP and 3KP). In 1KP, only A has to sign some data, which must be verified by C and M.

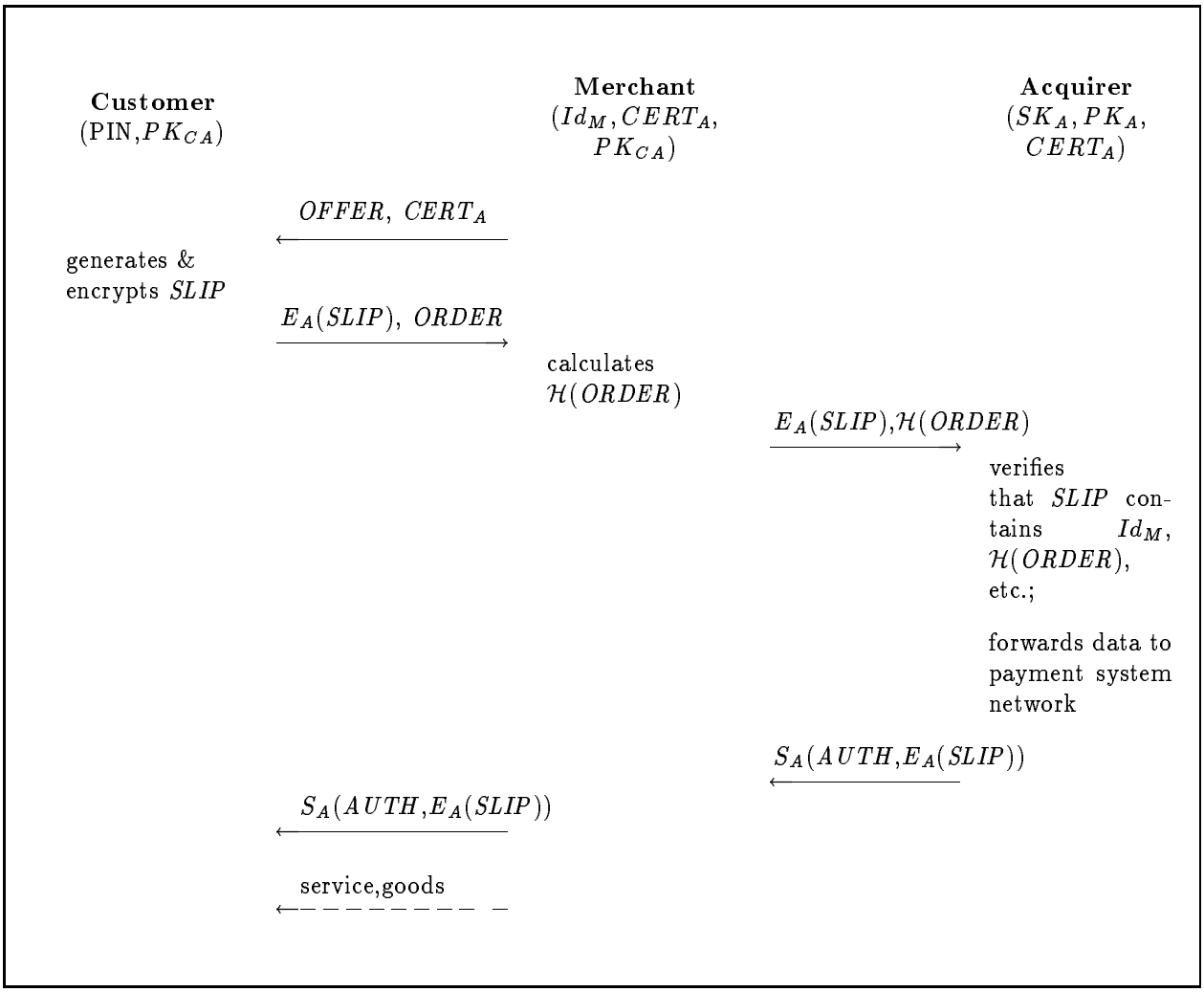

Figure 3: 1KP Protocol

The values used in the protocols are defined as follows:

- *OFFER*
  - offer description,
  - amount, currency, date, ID of merchant
- *ORDER*
  - order description,
  - amount, currency, date, ID of merchant,
  - delivery address (electronic or physical)
- *SLIP*
  - amount, currency, date, ID of merchant,

  - credit card number, expiration date, PIN,
  - $\mathcal{H}(ORDER)$

- $AUTH$
  - 'approved'/'rejected'
  - $\mathcal{H}$(amount, currency, date, ID of merchant)
  - $\mathcal{H}(ORDER)$

When the customer places an order (flow not shown), the merchant responds with the full information: $OFFER$, and a certification $CERT_A$ for A's public key. The customer checks the validity of $CERT_A$. He then forms $SLIP$ and encrypts it under A's public key, via the function E_A, to get a ciphertext $y = E_A(SLIP)$. He transmits y and $ORDER$ to the merchant.

The merchant checks that $ORDER$ matches his offer. He then computes $h = \mathcal{H}(ORDER)$, and forwards h, y to the acquirer gateway.

The acquirer gateway decrypts y. If the decryption fails, then the alteration of y is detected and the transaction is invalid. If not, A gets $SLIP$. Now A extracts $\mathcal{H}(ORDER)$ and ID of merchant from $SLIP$ and checks that these match the value h and the ID sent by the merchant. Using amount, currency, date, credit card number, expiration date, and PIN from $SLIP$, A uses today's existing clearing and authorization system to *on-line* authorize the payment.

Upon receipt of a response from the authorization system, A computes a signature, using the function S_A, on $AUTH$ and $y = E_A(SLIP)$. If the payment is authorized, $AUTH$ includes the value 'approved', otherwise the value 'rejected'. The signature is then transmitted to the merchant.

The merchant checks the validity of the signature, based on the already known data amount, currency, date, ID of merchant, h, and y. If the signature is valid and contains the 'approved' value, M forwards the signed authorization to the customer.

We stress here that the use of $\mathcal{H}$(amount, currency, date, ID of merchant) in the field $AUTH$ of the signature (as opposed to using the explicit values amount, currency, etc.), is done in order to protect the privacy of these data when transmitted to merchant and customer.

1KP satisfies the following requirements:

- A1.a: *Proof of Transaction Authorization by Customer.* $SLIP$ includes the PIN, which is known only to the payment system (especially, A) and the customer C. Since C knows PK_{CA} and verifies $CERT_A$, it is ensured that C does not send the PIN to any non-authorized party. A decrypts and checks that the PIN is correct. The plaintext-awareness of the encryption (see beginning of Section 5) implies that $SLIP$ originated with the PIN-holder. Thus, an adversary not knowing the PIN can neither create a fake $SLIP$ nor modify the encryption of a legitimate one to its advantage.

  Since $SLIP$ includes a timestamp, replay by a dishonest merchant can be detected. Either the clearing network handles replay detection or A must keep state about past timestamps (or transaction identifiers) for the duration of the "acceptable delay period."

  The plaintext-awareness (or more specifically, the fact that E_A is randomized) implies also security against *dictionary-attacks*: If the attacker knows all data in $SLIP$ except

PIN, he could compute encryptions $E_A(SLIP)$ for *all* possible values of PIN. With a deterministic encryption function, he could easily determine the correct PIN by comparing all encryptions with that one produced by C. Therefore, plain-text aware encryption is randomized: if *SLIP* is encrypted twice, two *different* cyphertexts are produced, which excludes this type of attack.

Note that PIN-based authentication provides a weak proof only. Signature-based authentication as used in 3KP provides an undeniable proof. Moreover, the probability of guessing the correct PIN is much higher than the probability of guessing a valid-looking signature.

- M1.b: *Proof of Transaction Authorization by Acquirer.* The unforgeable, undeniable proof is the digitally-signed message sent by A. Notice that we have used a digital signature so that non-repudiability is provided. The inclusion of $\mathcal{H}(ORDER)$ prevents the replay of authorization messages which would result in fake authorization of customer's orders.

  Since the merchant knows $AUTH$ in advance, the signature would indicate any tampering in the information sent from merchant to acquirer, and any disagreement between customer and merchant on the payment data.

  The inclusion of $\mathcal{H}(ORDER)$ both in the customer-generated *SLIP* and in the flow from merchant to acquirer enables A to detect a disagreement between merchant and customer with respect to the order contents (even before submitting the transaction to the clearing network).

- C1.a: *Unauthorized Payment is Impossible.* This is a direct consequence of A1.a.

  Notice that A's authorization sent to the merchant includes $\mathcal{H}(ORDER)$ as provided by C in *SLIP*, which authenticates the payment data including the *delivery address*. This prevents a certain kind of *man-in-the-middle* attack that we now describe:

  An attacker that impersonates a merchant can get the agreement of the customer to buy something for a given amount. The adversary gets from the customer an encrypted slip authorizing the payment. The adversary now impersonates the customer to the merchant, but this time the adversary buys for the same amount a (possibly) different merchandise with different delivery address and "pays" for it with the customer's slip. Notice, however, that in this case there will be a mismatch between the view of the "order" by the real customer and the merchant, and, consequently, a mismatch in the value of $\mathcal{H}(ORDER)$.

- C2.b: *Proof of Transaction Authorization by Acquirer.* As for M1.b.

We note that the last flow from merchant to customer in which the signed authorization by the acquirer is transmitted is optional. It only serves as a receipt for the customer but is not needed for the security of the payment protocol.

To summarize, 1KP is a simple and efficient protocol whose main achievement is to get a secure electronic payment system with as little modification as possible to the existing infrastructure. Its main weaknesses are: 1) the customer authenticates itself via the acquirer and only using a credit card number and PIN (as opposed to a strong authentication via a digital signature); 2) the merchant does not directly authenticate itself to the customer or acquirer (there is some level of indirect authentication via the customer's *SLIP* and the authorization by the acquirer); and 3) neither merchant nor customer provide undeniable

receipts for the transaction. Upgrading 1KP to provide these missing features results in the protocols described in the next two subsections, namely, 2KP and 3KP.

5.2 2KP

The second protocol, 2KP, is illustrated in Figure 4. The basic difference with respect to 1KP is that, in addition to A, each merchant M needs to possess a public key with a matching secret key, and distribute its own public key, with its certificate, $CERT_M$.

We require that $CERT_M$ also include the ID of merchant, ID_M.

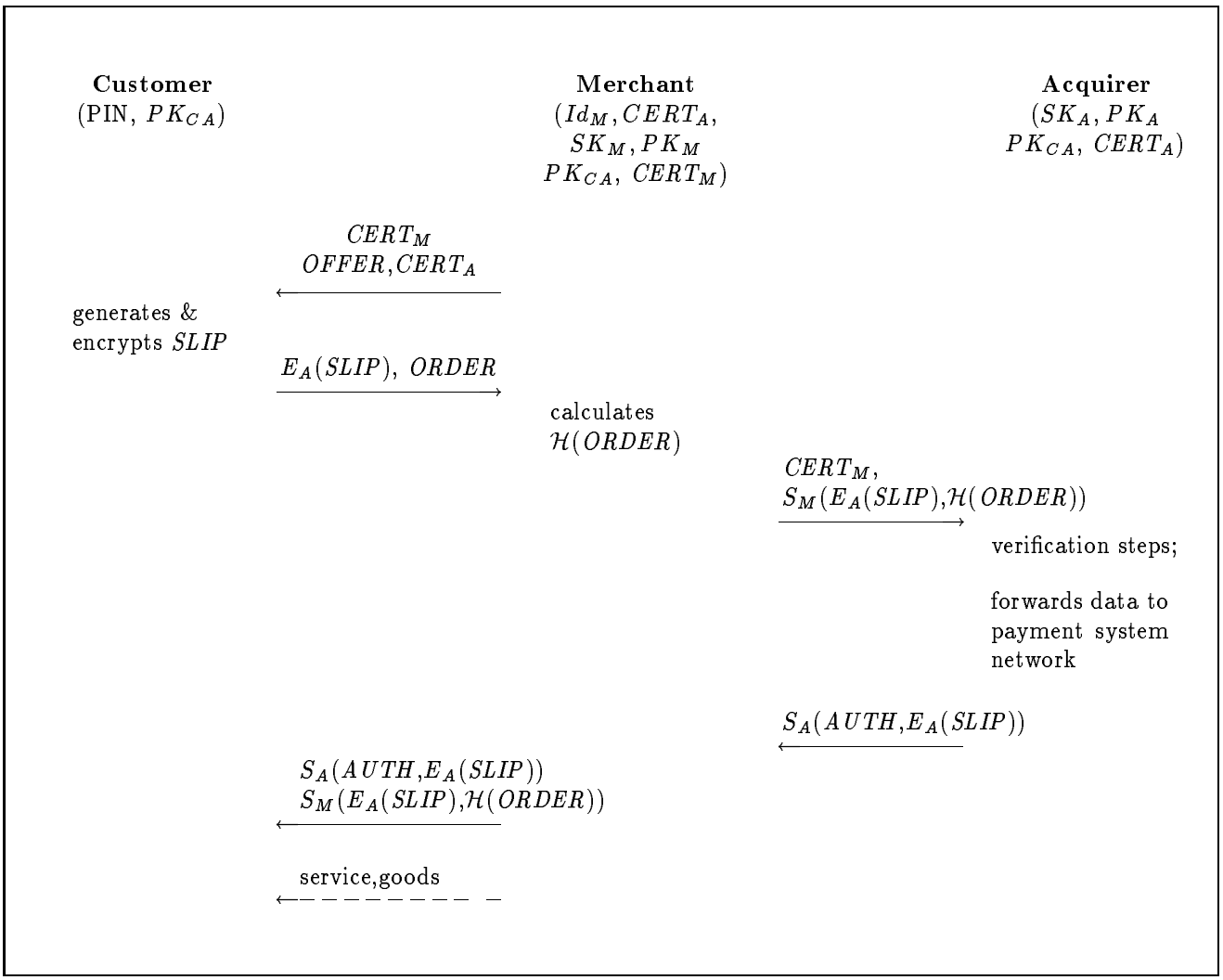

Figure 4: 2KP Protocol

2KP satisfies all the requirements addressed by 1KP as well as:

- A2: *Certification and Authentication of Merchant.* This is achieved by $CERT_M$ and M signing its flow to the acquirer.

- C3: *Certification and Authentication of Merchant.* This is achieved by $CERT_M$, which includes the merchant's identity.
- C4: *Receipt from merchant.* This is transitively achieved by M's signed message sent to A and A's signed authorization message. Obviously, M can refuse forwarding A's authorization message to the customer and sending its last message. In this case, C does not know whether the transaction was aborted or finalised (this must be handled based on the next statement of account).

  The same could be achieved by M signing A's authorization message, but at the cost of an additional signature.

5.3 3KP

As can be expected, in the last protocol – 3KP – all protocol participants, including customers, possess a public key, with the associated secret key and certificate. As illustrated in Figure 5, all parties are now able to provide non-repudiation.

The $CERT_C$ is sent to the merchant, it may not only contain the customer's public key and ID, but also further data. This further data is included in the certificate *in hashed form* in order to have to reveal it to the merchant only on demand. For instance, $CERT_C$ might include $\mathcal{H}$('Customer's physical address'), and if ordered goods should be sent to C's home address, C can reveal 'Customer's physical address' to the merchant who can verify it based on $CERT_C$.

The customer's signature serves as undeniable proof of transaction (A1.b), and enables disputability (C1.b). On the other hand, the merchants can link all payments of the customer with $CERT_C$ and C's signature, i.e., the customer loses some of the privacy compared to 1KP and 2KP. One way to avoid this is by encrypting $CERT_C$ and the signature with A's public key.

The merchant forwards C's signature to the acquirer, and authenticates this message by signing it under S_M. Note that, in contrast to 2KP, the merchant is not explicitly signing the value of $\mathcal{H}(ORDER)$. Since this value is included under C's signature, and M can verify it, then it suffices for M to sign only C's signature. In the case that the customer's signature is not revealed to M (for privacy), the flow from the merchant to the acquirer must include $\mathcal{H}(ORDER)$ as computed by the merchant, like in 2KP.

Notice that in 3KP the use of PIN numbers is only for compatibility with the existent infrastructure. Except for that reason, PINs can be safely omitted since the level of authentication provided by the customer signature is significantly superior to that provided by a PIN.

3KP satisfies all the requirements addressed by 2KP as well as:

- A1.b: *Undeniable Proof of Transaction Authorization by Customer.* The customer signs the *SLIP* using a secret key SK_C known to C only.
- M2.b: *Proof of Transaction Authorization by Customer.* Based on C's signature, M can verify that *SLIP* was signed by C. M cannot verify the correctness of the contents of *SLIP*, especially not of the PIN.
- C1.b: *Unauthorized Payment is Impossible.* Follows from A1.b.

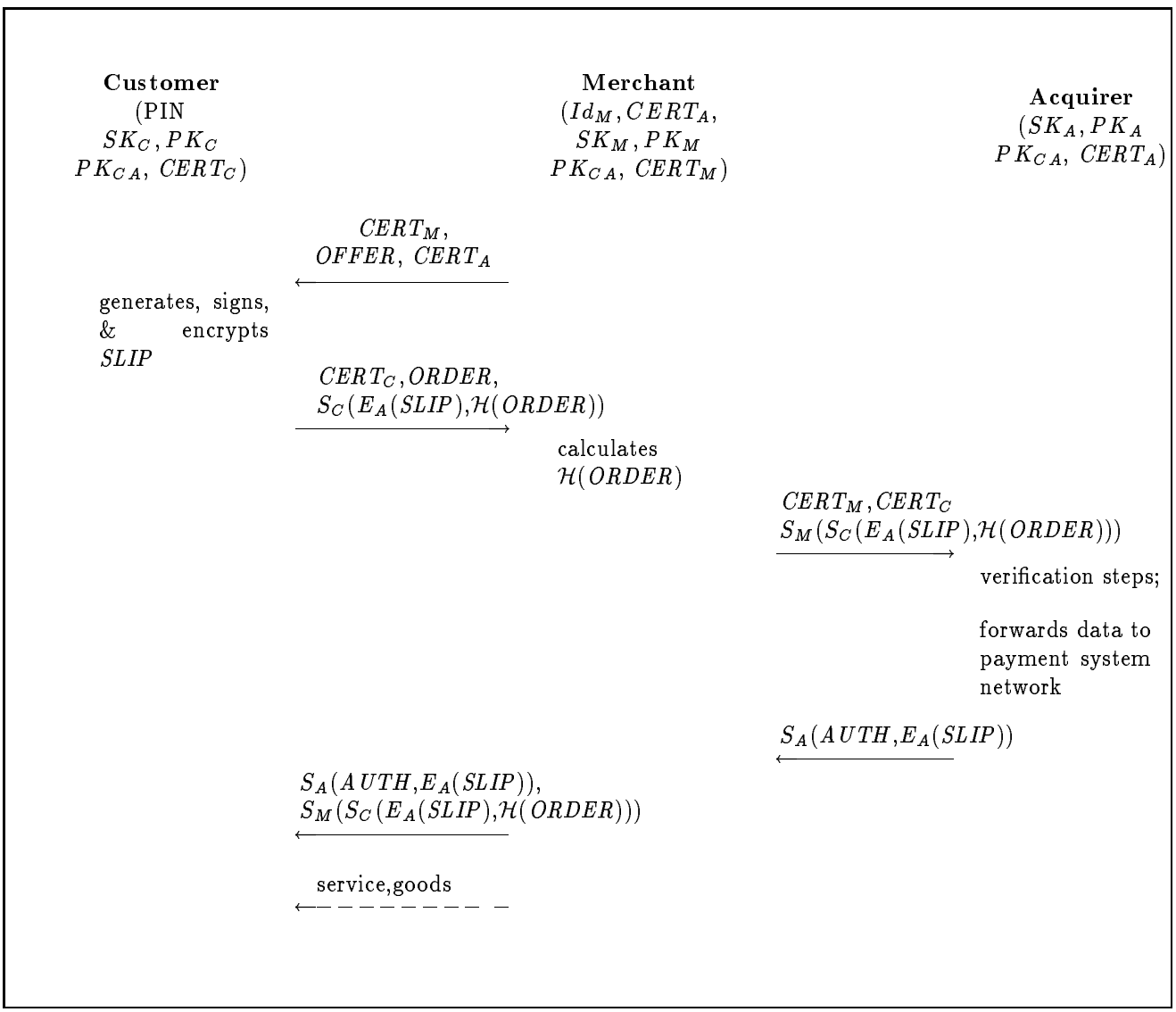

Figure 5: 3KP Protocol

5.4 Certificate Distribution

The protocol in Figure 6 shows how certificates of merchants and customers can be generated and distributed. This is only a simplified version as there is one acquirer only, and besides Id_X and PK_X, the contents of $CERT_X$ are not defined (see Section 3.2).

All certificates required to certify a particular public key will be provided together with the key so as to keep the state information of merchants and acquirers at a minimal level. Since certificates of merchants and customers are checked in real time by acquirers their revocation is not a problem. Revocation of acquirers can be alleviated by having an expiry on their certificates. Revocation of the key of the certification authority of course is a major problem.

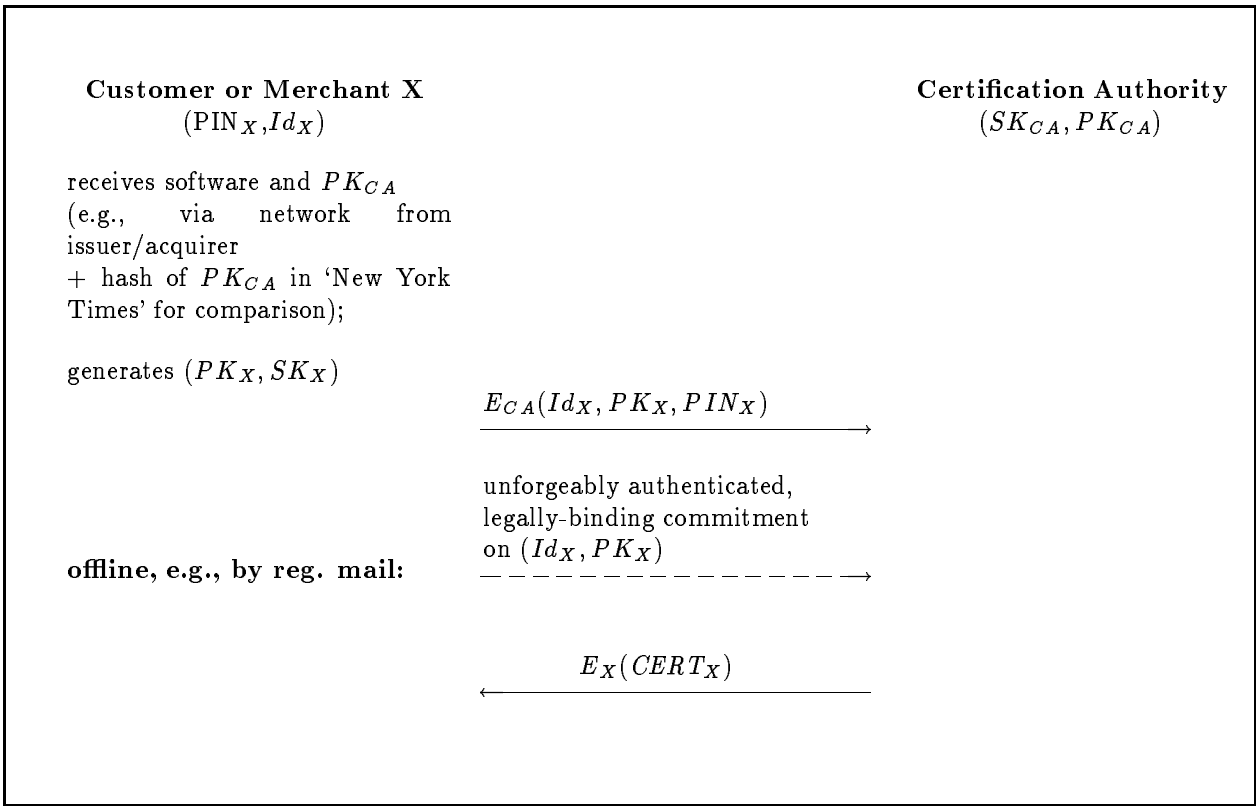

Figure 6: Certificate Distribution

Noteworthy is that the customer and the merchant generate their own keys independently:

- Technically it is not a problem (Customer and merchant use a computer and satisfactorily random input can be derived from measuring the intervals between keystrokes);
- contrary to the generation of PIN's, weak keys can be generated only with difficulty on purpose; and
- since $S_X(\cdot)$ should provide non-repudiation towards the payment system, SK_X must not be known to A.

If more assurance is needed one can use two keys, one generated by the customer or merchant, and one by the certification authority.

6 Summary and Comparison of the Protocols

The *i*KP protocols vary in the degree of both protection and complexity. They proceed in an incremental path towards electronic payment with strong security features with respect to all parties involved. Practically speaking, it is envisaged that 1KP will represent a short-term, interim step towards payment protocols with stronger security guarantees. Thereafter, 2KP and 3KP can be gradually phased in. Table 1 presents a comparison of the *i*KP protocols.

REQUIREMENTS/PROTOCOLS	1KP	2KP	3KP
Issuer/Acquirer			
A1. Proof of Transaction Authorization by Customer	√	√	√√
A2. Proof of Transaction Authorization by Merchant		√√	√√
Merchant			
M1. Proof of Transaction Authorization by Acquirer	√√	√√	√√
M2. Proof of Transaction Authorization by Customer			√√
Customer			
C1. Unauthorized Payment is Impossible	√	√	√√
C2. Proof of Transaction Authorization by Acquirer	√√	√√	√√
C3. Certification and Authentication of Merchant		√√	√√
C4. Receipt from Merchant		√√	√√

Table 1: Comparison of the *i*KP payment protocols. A requirement marked by √ is satisfied but not disputable, while √√ indicates that the requirement is satisfied based on an undeniable proof, providing non-repudiation and disputability.

The *i*KP family can fulfill all stated requirements and, in particular, provide non-repudiatable receipts from the acquirer gateway to the merchant/customer, and from the merchant to the customer. In case that the customer also possesses a public-key pair (3KP), non-repudiation becomes possible also from the customer to the merchant/acquirer.

The protocols do not reveal the identity of the customer to the merchant. Order privacy against eavesdroppers could be achieved by applying a secure communication protocol (e.g., SHTTP [18] or SSL [14]), or, if desired, the *i*KP protocols themselves could be extended to provide that protection. Since *i*KP aims at credit-card-like payments, no anonymity against the payment system is provided. Adding anonymity and privacy to all payments is a major change in "payment culture" and only after the deployment of *i*KP-like systems, it will be assessable whether the involved parties are inclined to move further into this direction.

The *i*KP protocols can be extended to support batch processing of payments from the same customer by the merchant, or to guarantee amounts as commonly done, for example, in the case of car rentals.

The protection of the acquirer from the Internet is another important aspect to the acceptability of such payment systems - first designs to minimize this exposure have been accomplished.

References

[1] R. Anderson. Why Cryptosystems Fail. *Communications of the ACM 37/11 (1994) 32-41.* <ftp://ftp.cl.cam.ac.uk/users/rja14/wcf.ps.Z>

[2] M. Bellare, P. Rogaway. Optimal Asymmetric Encryption. *Pre-proceeding of Eurocrypt '94*, May 1994, University of Perugia, Italy, 103-113.

[3] J.-P. Boly *et al.* The ESPRIT Project CAFE - High Security Digital Payment Systems. *ESORICS '94*, LNCS 875, Springer-Verlag, Berlin 1994, 217-230.

<http://www.informatik.uni-hildesheim.de/FB4/Projekte/sirene/projects/-cafe/index.html>

[4] H. Bürk, A. Pfitzmann. Digital Payment Systems Enabling Security and Unobservability. *Computers & Security*, 8/5 (1989), 399-416. <http://www.informatik.-uni-hildesheim.de/FB4/Projekte/sirene/lit/abstr89.html#BuePf_89">

[5] D. Chaum. Privacy Protected Payments. *SMARTCARD 2000, Proceedings, North-Holland, Amsterdam 1989, 69-93.*

[6] D. Chaum. Achieving Electronic Privacy. *Scientific American*, August 1992, 96-101.

[7] P. Cheng, J. Garay, A. Herzberg, and H. Krawczyk. Design and implementation of modular key management protocol and IP Secure Tunnel on AIX. In *Proc. 5th USENIX UNIX Security Syposium*, Salt Lake City, Utah, June 1995.

[8] W. R. Cheswick, S. M. Bellovin. Firewalls and Internet Security: Repelling the Willy Hacker. *Addison Wesley, 1994.*

[9] CyberCash. The CyberCash(tm) System - How it Works. <http://www.-cybercash.com/cybercash/cyber2.html>

[10] DigiCash. About ecash. <http://www.digicash.com/ecash/ecash-home.html>

[11] S. Dukach. SNPP: A Simple Network Payment Protocol. *Computer Security Applications Conference,1992.* <ftp://ana.lcs.mit.edu/pub/snpp/snpp-paper.ps>

[12] D. K. Gifford, L. C. Stewart, A. C. Payne,G. W. Treese. Switches for Open Networks. *IEEE COMPCON*, March 95. <http://www.openmarket.com/-about/technical/>

[13] R. Hauser, M. Steiner. Generic Function Extension of WWW Browsers. *IBM Research Division, Zurich Research Lab, March 1995.*

[14] K. E. B. Hickman. Secure Socket Library. *Netscape Communications Corp., Feb. 9th, 1995* <http://www.mcom.com/info/SSL.html>

[15] P. Janson, M. Waidner. Electronic Payment over Open Networks. *Zeitschrift der Schweizerischen Informatikorganisationen xxxx (1995) xxxx-xxxx.* <http://www.-zurich.ibm.ch/Technology/Security/extern/ecommerce/>

[16] S. H. Low, N. F. Maxemchuk, S. Paul. Anonymous Credit Cards. *2nd ACM Conference on Computer and Communication Security, Fairfax 1994.* <http://www.-research.att.com/index.html#acc>

[17] C. Neuman, G. Medvinsky. Requirements for Network Payment: The NetCheque Perspective. *IEEE COMPCON*, March 95. <ftp://prospero.isi.edu/-pub/papers/security/netcheque-requirements-compcon95.ps.Z>

[18] E. Rescorla, A. Schiffman. The Secure HyperText Transfer Protocol. Internet Draft. *Enterprise Integration Technologies, December 1994.* <http://www.eit.com/-projects/s-http/>

[19] B. Schneier. Applied Cryptography. *John Wiley & Sons, 1994.*

[20] M. Sirbu, J. D. Tygar. NetBill: An Internet Commerce System. *IEEE COMPCON, March 95.* <http://www.ini.cmu.edu/netbill/CompCon.html>

[21] L. H. Stein, E. A. Stefferud, N. S. Borenstein, M. T. Rose. The Green Commerce Model. *First Virtual Holdings Incorporated, October, 1994.* <http://www.fv.com/tech/green-model.html>