Document #	211ZR018	Activity Paper	
Title	Electronic Payment Systems		
Author / Body	Phil Janson, Michael Waidner		
Editor	M. Waidner (ZRL)		
Date	May 1st, 1996		
Status	Final	Vers. 8	public
CEC / TA id	none		

SEMPER

Activity Paper

211ZR018

May 1st, 1996

Electronic Payment Systems[1]

Phil Janson, Michael Waidner

May 1st, 1996

ABSTRACT

As business is moving from face-to-face trading, mail order and telephone order to electronic commerce over open networks such as the Internet, crucial security issues are being raised. While Electronic Funds Transfer over financial networks is reasonably secure, securing payments over open networks connecting commercial servers and consumer workstations poses challenges of a new dimension.

This report reviews the state of the art in payment technologies, and sketches emerging developments, especially IBM's Internet Keyed Payment protocol, *iKP*.

1 Introduction

Since the dawn of history there has been trading between two parties exchanging goods face-to-face. Eventually such trading became complicated and money was invented so a buyer could acquire something he needed from a seller without necessarily exchanging goods. Security of the monetary systems was guaranteed by the local, regional, national and eventually international banks controlling the printing of money. New steps were taken when payment orders, cheques, and later 'plastic' money were invented. This allowed payment without 'actual' money, the mapping between the payment instrument and the real money being still guaranteed by the banks through secure financial clearing networks. Eventually remote payment became possible using those same instruments, although security then started to become a challenge. Indeed verifying a signature on a cheque or a credit card mail order when buyer and seller are not face-to-face is impossible: either the buyer must take the risk of sending in a payment before having received his purchase or the seller must send the purchased items before having received the payment. Phone order purchases are even worse since no signature at all can be provided: the seller runs the risk that the buyer may deny having made the purchase and demand a refund even after he has received the goods. Without new security measures, envisioning electronic commerce over open computer networks is utopic.

This report discusses electronic payment models in Section 2, security requirements in Section 3, and emerging technologies in Section 4. In Section 5 we describe *iKP* as the prototype of a credit-card based Internet payment system. The status of standardisation activities for this kind of payment systems is summarised in Section 6.

---

1 Sections 1-4 and 7 of this report are based on [JaWa 95]. This work was partially supported by the ACTS Project AC026, *SEMPER*; however, it represents the view of the authors. *SEMPER* is part of the *Advanced Communication Technologies and Services* (ACTS) research program established by the *European Commission Directorate General XIII*.

2 Electronic Payment Models

Commerce involves always a **buyer** and a **seller** (or **payer** and **payee**) who need to exchange money for goods or services, and at least one **financial institution** which links “bits” to “money.” In most existing payment systems, this role is divided into two parts, an **issuer** (used by the buyer) and an **acquirer** (used by the seller).

The electronic payment from buyer to seller is always implemented by a flow of real money from the buyer via the issuer and acquirer to the seller.

In **pre-paid “cash-like” payment systems**, the buyer’s bank account is debited before any purchase is made, by a certain amount which can be used for payments later. Card-based electronic purses, electronic cash as well as (certified/guaranteed) bank cheques fall in this category. The typical flows are shown in Figure 1.

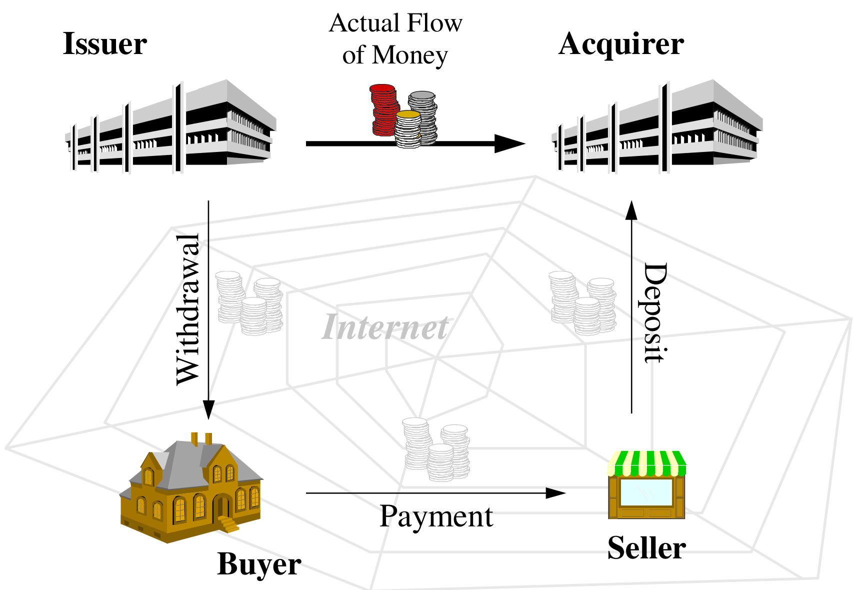

Figure 1 Cash-like Payment System

In **pay-now** payment systems, the buyer’s account is debited at the time of payment. ATM-card based systems (like the European *EC-Direct* system, *edc*) fall in this category. In **pay-later (credit)** payment systems, the seller’s bank account is credited the amount of the sale before the buyer’s account is debited. Credit card systems fall in this category. From a protocol point of view pay-now and pay-later systems belong to the same class. The typical flows of these systems are shown in Figure 2. Since a payment is always done by sending some sort of “form” from buyer to seller (cheque, credit card slip, etc.) we call these systems **“cheque-like”.**

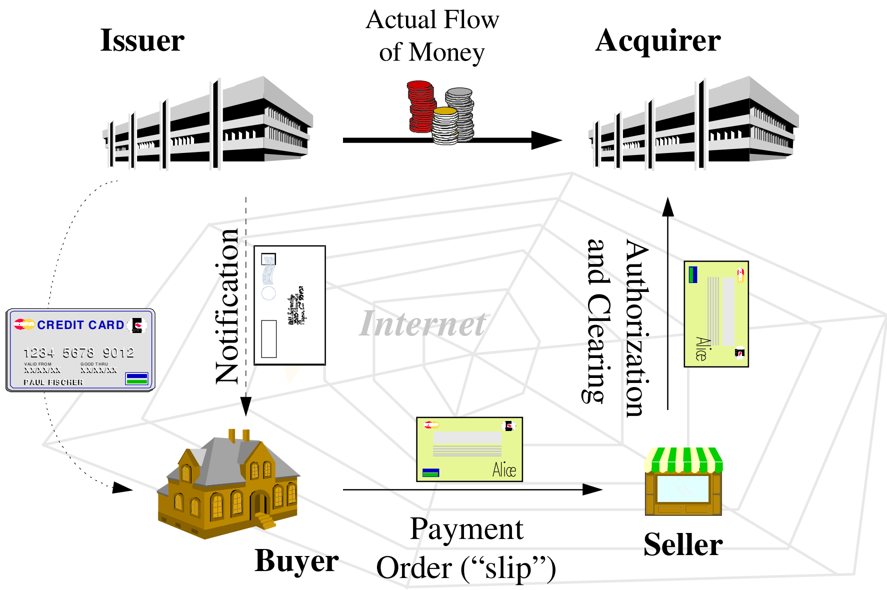

Figure 2 Cheque-like Payment System

Both types of payment systems are *direct* payment systems, i.e., a payment requires an interaction between buyer and seller. There are also *indirect* payment systems where either buyer or seller initiates the payment without having the other party (seller or buyer, respectively) involved online. In the context of Internet payments this is usually considered as part of "home banking."

3 Security Requirements

Except for the high level requirement that nobody wants to lose money, and everybody wants to be unobservable in their business to some extent, the concrete security requirements of electronic payment systems vary depending on their features and the trust assumptions put on their operation. Generally, though, one or more of the following requirements must be met.

3.1 Integrity and Authentication

For a payment system, integrity means that no party loses money unless a payment is explicitly authorised by that party. Additionally a party might require to not receive any money without their explicit consent, e.g., in order to prevent unwanted bribery.

Technically this is achieved by *authenticating* those messages that cause the transfer of money (withdrawal/payment/deposit in cash-like systems, payment/clearing in cheque-like systems). There are several possibilities how to do this:

- **Out-band Authorisation:** The verifying party sends a notification to the authorising party and requires that the authorising party approves or denies the authorisation offline, using a secure out-band channel (e.g., ordinary mail or phone calls).

This is the current approach for credit card payments in mail-orders and telephone-orders (MOTO): The transaction is performed without signature, i.e., everybody who knows a buyer's credit card data can initiate such payments. The buyer must check their credit card statements and must actively complain about fake transactions. Non-reaction within a certain time (usually 90 days) is interpreted as "approved" by default.

The obvious problems are the delay caused by the out-band communication, and that the authorising party must carefully check the notifications, especially if "approved" is the default value as in credit card payments.

- **Authorisation by passphrase:** The verifying party requires that messages from the authorising party include a cryptographic check value that is computed using a secret known to the authorising and verifying party only, e.g., a PIN or passphrase, or in general a "shared secret."[2]

  This achieves security against outsiders, but a dispute between authorising and verifying party about the origin of a well-authenticated message cannot be resolved (i.e., there is no non-repudiation of origin). This means that, e.g., a court would not be able to decide from the messages exchanged whether a disputed payment was authenticated by the buyer or by dishonest employees of the buyer's bank. Typically such insider fraud requires employee *collusion* since in PIN-based systems, customer PINs are usually well protected through encryption and tamper-resistant hardware at the bank.

  Additionally, short shared secrets like PINs of 4 or 6 digits cannot provide a high degree of security by definition.[3] Therefore short shared secrets should only be used to control access to a physical token like a smartcard (or wallet, see Section 4.2), and this token should be used to do the actual authorisation based on really secure cryptographic mechanisms.

- **Authorisation by signing:** The verifying party requires a digital signature of the authorising party.

  Digital signatures provide non-repudiation of origin: Only the owner of the secret signature key can sign messages (while everybody who knows the corresponding public verification key can test signatures). This enables to resolve disputes in case the authorising and verifying party disagree about the authenticity of a message.

Additionally the *freshness*[4] of authenticated messages must be ensured. This is achieved by including *time stamps*, *sequence numbers,* or *nonces*[5].

---

2 In cryptology this is called *symmetric authentication*, and the check value is called *MAC*, for *message authentication code.*

3 The probability of guessing a correct PIN is already high, especially for user-chosen PINs (which are likely to be related to the user's birthday, phone number, etc.). In the worst case the attacker can perform a so-called *dictionary attack*: If a value X is known that is a deterministic function of the PIN, one can use this X to search the set of all possible PINs for the correct one. In some applications one can prevent dictionary attacks by randomization [BeMe 92].

4 *Freshness* means that the message provably belongs to the current context only (e.g., the current payment transaction), i.e., that it is no *replay* of a previous message.

5 A *nonce* is a random value chosen by the verifying party and sent to the authenticating party. Since nonces are unpredictable and used only in one context (e.g., one payment transaction), they ensure that a message cannot be reused in later transactions. Since nonces do not require

Section 4.3 lists some examples of payment systems structured according to the technique used for authorising the buyer's transfer order towards the bank (which is the most important relation in a payment system).

In addition to these requirements, buyer and seller might want some fairness in the exchange of payment for receipt, or more directly in the exchange of money for goods and services (e.g., by means of a trusted third party). Usually this is out of scope of payment systems.

3.2 Confidentiality

Some or all parties involved may wish confidentiality of the transaction. Thereby the knowledge of the identity of buyer, seller, purchase content, amount, etc., are restricted to the participants involved, or even only to a subset of them (e.g., where anonymity or untraceability are desired). See Section 4.4.

3.3 Availability and Reliability

All parties are interested in being able to perform or receive payments whenever necessary.

Payment transactions must be atomic, i.e., happen entirely or not at all, but never hang in an unknown or inconsistent state. No buyer would accept to lose money (or at least no significant amount) due to a network crash, or because the seller's server crashed.

Availability and reliability presuppose that the underlying networking services and all software and hardware components are sufficiently dependable. Recovery from crash failures requires some sort of stable storage at all parties and specific resynchronisation protocols. These fault tolerance issues are not discussed in the following, since most payment systems do not address them explicitly.

3.4 Requirements for Mobile Trusted Hardware

To address the security requirements, digital signatures and secret key distribution functions are required, which makes it desirable that all parties involved have access to secure key storage. If payments are to be possible from any workstation, the secret key storage of a user must even be mobile, which in turn makes it necessary that users be provided with smart cards or similar secure devices. These technology options are discussed in the following Section 4, together with some examples.

synchronisation between the two parties they are very robust, and popular in cryptographic protocol design. In general, *nonces* are an example of a *challenge-response* technique [Schn 95].

4 Technology Overview

	Online Payment Systems	**Offline Payment Systems**
Systems enabling control of payment flow by payment system provider	**Credit-card Payment Systems:** • **Proposal using no Cryptography:** *First Virtual* [SSBR 95, Bore 95] • **Proposals using Cryptography:** *CyberCash* [East 95, EcBY 95], *iKP* [BGHH 95, LiTs 95] • **Proposed Standard:** *SET* [SET 96] **Micropayments:** • *Millicent* [Mana 95], *NetBill* [SiTy 95], *Phone-Ticks* [Pede 95], *iKP-Micropayments* [HaSW 95], *Pay-Word* [RiSh 95]	**Electronic purses, using smart cards:** • **Shared key,** e.g., *Danmont* (DK), *Proton* (B) • **Public key,** e.g., *Express* (B) • **Not known publicly:** *Mondex* (UK) [Rolf 94][6] • **Standardisation:** *CEN Intersector Electronic Purse, EMV Electronic Purse.* Suggested for online payments. Loading cash via smartphone[7] possible.
Untraceable Systems	• **Anonymous "remailers" for change,** e.g., *NetCash* [MeNe 93, NeMe 95, Neum 95], *Anonymous Creditcards* [LoMP 94] • **Anonymous ("blind") signatures,** *e-cash* [Chau 89]	• **Anonymous ("blind") signatures,** e.g., *CAFE* (research project, sponsored by the European Union) [BBCM 94]. Suggested for online payments, but not applied yet. Loading cash via smartphone possible.

Figure 3 Proposed Technologies for *Internet* Payments

The main design decision for electronic payment systems is how to authorise payments, i.e., how to enable the honest buyer to convince the seller to accept a legitimate payment while preventing the dishonest buyer from doing unauthorised payments. For instance, it must not be possible to spend the same money twice by sending the same messages to two different sellers. And all this must be done in a way that does not harm the privacy of honest buyers and sellers.

4.1 On-line vs. Off-line

Payments can be performed **on-line,** involving an authentication and authorisation server (usually as part of the issuer or acquirer) in each payment, or **off-line,** i.e., without contacting a third party during the payment from buyer to seller.

6 The design of *Mondex* is not public. According to presentations by *Mondex* they are using public key technology, but it is not known where and for what purpose. The actual hardware they are using suggests that the system is primarily a shared-key system.

7 A "smartphone" is a telephone with smartcard reader and display.

The obvious problem with off-line payments is how to prevent buyers from spending more money than they actually possess. In a pure digital world there is nothing that would prevent a dishonest buyer from resetting the local state of their system after each payment to the state before the payment.

Therefore offline payment systems that *prevent* doublespending (i.e., do not just ensure that doublespending is detected after it happened) require tamper-resistant hardware at the buyer, e.g., smartcards [e.g., GuUQ 92]. Often tamper-resistant hardware is also used at the seller, e.g., security modules of POS-terminals. It is required for shared-key systems, and in case the seller does not forward the individual transactions but their totals only.

On-line systems require, of course, more communication, but not necessarily specific tamper-resistant hardware at the buyer and seller. In general, they are considered as more secure than off-line systems. Most proposed Internet payment systems are on-line systems.

All proposed electronic purses are offline systems, e.g., *Danmont, Proton* (by Banksys), *Mondex* [Rolf 94], *Express* (by Europay), *CAFE* [BBCM 94] (developed by a European ESPRIT project). The "exotic" systems in this list are *Mondex* and *CAFE*. *Mondex* is the only system that enables off-line transferability (i.e., the seller can use the amount received for a new payment, without intermediate deposit — but this seems to be a politically unpopular feature in most countries), and it is the only system considered for high-value payments (i.e., beyond 100 ECU). *CAFE* is the only system that provides strong buyer anonymity and untraceability. Both systems offer the buyers an electronic wallet, preventing the well-known fake-terminal attacks on the buyer's PIN. *Mondex, Express,* and *CAFE* are multi-currency purses, i.e., they can handle different currencies simultaneously.

All these systems could be used for Internet payments, but currently, they aren't. Plans exist for several systems, e.g., *Mondex* and *Express*. The main technical obstacle is that they require a smartcard reader attached to the buyer's PC or workstation. Inexpensive PCMCIA smartcard readers and standardised infrared-interfaces on notebook computers will solve this connectivity problem.

Another system that will is developed in this spirit is the *FSTC Electronic Check Project* that uses a tamper-resistant PCMCIA card, and implements a cheque-like payment model. (The related *FSTC Electronic Commerce Project* [Seel 94] deals with unifying the different payment methods at the acquirer's site.)

Alternatively to tamper-resistant hardware, off-line authorisation could be done via pre-authorisation, i.e., the seller is known to the buyer in advance, and the payment is already authorised during withdrawal, in a way similar to a certified bank cheque.

4.2 Trusted Hardware

As already mentioned in Section 4.1, for the issuer's security, most off-line payment systems require a piece of tamper-resistant hardware in the buyer's device. In a certain sense, this piece is a "pocket branch" of a bank and must be trusted by the issuer.

Independent of the issuer's security considerations, it is in the buyer's interest to have a secure device that the buyer can trust, i.e., that protects the buyer's secret keys and performs the necessary operations. Initially, this could be just a smartcard, but in the long run, it could become a smart device of a different form factor with secure access to some keyboard and display for entering a PIN and simple commands (for a more detailed discussion see [PPSW 95]). This is often called an *electronic wallet.*

Without such a secure device, the buyer's secrets, i.e., the buyer's money, are vulnerable to anybody who can access the buyer's machine. This is obviously a problem in multi-user environments, but also on single-user machines that may be accessed directly or indirectly by others, and where a virus, for instance, could steal PINs and passwords while they are entered.

Even with a smartcard, the buyer cannot be sure that a Trojan horse in his or her PC cannot reveal PINs of credit cards by sending mail to a remote attacker, or by simply asking the smartcard to make a payment to the attacker's account, silently. Therefore, for real security trusted input and output channels between user and "pocket workstation" must exist, e.g., using an electronic wallet with secure access to some display and keypad.

4.3 Cryptography

4.3.1 "Crypto-less" Systems

Using no cryptography at all means relying on "out-band" security: For instance, the goods ordered electronically are not delivered before a fax (or a letter, or a phone call) arrives from the buyer that acknowledges this order.

Examples for this kind of systems are *CompuServe, First Virtual* [Bore 95, SSBR 95], and the *Internet Shopping Network*. In *CompuServe,* each customer uses a line to dial into the system that is assumed to be secure by itself, and identifies himself or herself to the system. For both *First Virtual* and the *Internet Shopping Network,* the buyer has an account with the system and receives a password in exchange for a credit card number. The password is not protected while travelling over the Internet, i.e., these systems are vulnerable against eavesdropping. *First Virtual* has some protection by asking the supposed buyer for an acknowledgement of each payment via email, but the actual security of the system is based on the fact that buyers can revoke each payment within a certain time (i.e., there is no definite authorisation during the payment), and until then the seller takes the full risk.

4.3.2 Generic "Payment Switch"

A special role is played by the *OpenMarket Payment Switch* [GSPT 95]: It is an on-line payment system implementing both, a pre-paid and pay-later models. The architecture supports several authentication methods, depending on the payment method chosen, ranging from simple, unprotected PIN-based authentication to challenge-response based systems, where the response is computed, e.g., by a smartcard.

Actually, *OpenMarket* uses passwords and optionally two types of devices for response generation, *Secure Net Key* and *SecureID*, i.e., shared-key cryptography. (For authorisation, the *Payment Switch* digitally signs an authorisation message sent to the seller, i.e., public-key cryptography is used for this special purpose.)

4.3.3 Shared-key Cryptography

For shared-key cryptography, the buyer and seller on the one side and that party performing authentication or authorisation on the other side need a shared secret (e.g., a DES-key[8], or at least a password/PIN).

Since both sides have exactly the same information, this cannot provide non-repudiation (e.g., if buyer and issuer disagree about a certain payment, there is no way to decide whether this payment was initiated by the buyer, or by a dishonest employee of the issuer). Therefore authentication of the buyer's transfer order based on shared-keys is not appropriate if the buyer bears the risk of forged payments, at least not for high-value payments (say, greater than 100 ECU).[9]

If authentication should be done off-line (which is desirable at least for low-value payments), each pair of buyer and seller need a shared secret key. In practice this means that some sort of master-key is present at each seller, enabling the seller to derive the buyer's key, e.g., from the buyer's identity. Tamper-resistant security modules in point-of-sale terminals are used to protect this key.

Shared-key cryptography is used, e.g., by the off-line systems *Danmont* and *Proton,* and the on-line systems *SNPP* [Duka 91], *NetBill* [SiTy 95], and *NetCheque* [NeMe 95]. The *2KP* (a variant of *iKP*, see Section 5) may use a shared secret between buyer and issuer/acquirer for authentication.

The current versions of *NetBill* and *NetCheque* are based on the *Kerberos* technology, and focus on "micro-payments." Both implement a cheque-like debit payment model. The use of shared-key technology is justified by the performance required to process many micro-payments in short time. Both systems are announced to migrate to public key technology, and designs using public key technology exist.

4.3.4 Public-key Digital Signatures

For public-key cryptography, the buyer has a secret signature key and a certificate for their corresponding public signature verification key (e.g., issued by the payment system operator or a specific issuer). In most existing and proposed systems, RSA is used (but there are several alternatives).

Digital signatures provide non-repudiation, i.e., disputes between sender and recipient of a signed message can be resolved. This should be mandatory if the buyer bears the risk of forged payments, at least for high-value payments.

Off-line authentication is no problem here, since the seller can easily verify a signature of the buyer (and could check the certificate against a local copy of a blacklist of "bad" certificates, if necessary). Authorisation still requires either an on-line connection or trusted hardware at the buyer.

Public-key cryptography will be used, e.g., by the off-line systems *Express* and *CAFE*. Most of the proposed on-line payment systems use public key cryptography.

---

8 We do not provide references to cryptographic algorithms in this overview. For an introduction and a description of most of the cryptographic algorithms used for electronic payment systems, see [Schn 95].

9 For some examples on the consequences of no non-repudiation see [Ande 94].

Rather general WWW security schemes using public-key cryptography are *SHTTP* and *SSL*. Neither is a payment technology per se, but they are suggested for securing also payment information.

SSL [HiEl 95] is an Internet socket-layer communication interface allowing buyer and seller to communicate securely. [BLSS 95] describes an extension of *SSL*, called *PCT*. Neither *SSL* nor *PCT* support non-repudiation.

SHTTP [ReSc 95] is a secure extension of the *HTTP* protocol used on the Internet World-Wide Web (WWW), developed by the *CommerceNet* consortium. *SHTTP* offers several security techniques, e.g., signing and encrypting with RSA, on top of which various payment protocols could be implemented.

Both *SSL* and *SHTTP* offer a selection of symmetric and asymmetric cryptographic technologies. Because of the open access these technologies offer to strong cryptographic primitives, they are all subject to US export restrictions. Exportable versions will be available but they will be using 'tamed,' i.e., insecure cryptographic technologies. It is very questionable whether banks, sellers, and customers outside North America will be interested in such 'second-grade' technologies. Of course, since the specifications of these protocols are public, fully secure implementations may become available outside North America as soon as a non-US vendor decides to put such products on the market.

Complete payment systems using public-key cryptography are, e.g., *ecash* [Chau 89], *NetCash* [MeNe 93, NeMe 95], *Cybercash* [East 95, ECBY 95], *STT* [STT 95] and the *3KP* variants of *iKP* [BGHH 95, LiTs 95, vHST 96], *SEPP* [SEPP 95] and *SET* [SET 96].[10] *iKP* is described in Section 5.

4.4 Anonymity of Buyer

Some payment systems provide buyer anonymity and untraceability. Both are considered useful for cash-like payments (say, up to 100 ECU), with the argument that real cash is also anonymous and untraceable, and that buyers do not want to enable a third party to trace all their everyday-payments, certainly not the sellers (shops, publishers, etc.), but in some cases not even the banks.

While **anonymity** simply means that the buyer's identity is not used in payments, **untraceability** means that the buyer cannot be identified, and even that two different payments of the same buyer cannot be linked. Both properties hold with respect to the seller only or with respect to both seller and issuer/acquirer.

All payment systems could be made untraceable by outsiders, by encrypting all flows between buyer and seller, and anonymous to seller by using pseudonyms instead of real identities.

Some electronic payment systems are designed in a way which provides anonymity or even untraceability with respect to the seller (e.g., *iKP* offers this as one option).

---

[10] These are systems that somehow aim at real implementations. The protocol ideas themselves are much older: [Long 83] already describes how to use digital signatures for payments. [EvGY 83] did the same in an offline scenario (with "electronic wallets"), [Pfit 84] described anonymous accounts with digitally signed transfer orders. Even the ideas for ecash are quite old [Chau 83, Chau 89]. For early surveys see [BüPf 89, PWP 90].

Currently the only payment systems mentioned here that provide anonymity and untraceability against seller and issuer/acquirer are *ecash* [Chau 89] (on-line) and *CAFE* (off-line, based on smartcards), both based on asymmetric cryptography (a special form of signatures called blind signatures; for details see [Chau 89, Bran 94]).

NetCash [MeNe 93, NeMe 95] and *ACC* [LoMP 94] also provide anonymity and untraceability, but based on the use of trusted "mixes" [Chau 81] that change electronic money of one representation into another representation, without revealing the relation. Neither *ecash* nor *CAFE* assume the existence of such trusted third parties.

5 Internet Keyed Payment Protocol — *iKP*

IBM's Internet Keyed Payment Protocol (*iKP*) is a prototype of a credit-card based Internet payment system. It was part of *MasterCard's SEPP* proposal [SEPP 95], and contributed to the joint *MasterCard/VISA* proposal *SET* [SET 96].

iKP implements a cheque-like payment model (see Figure 2). The current specifications cover credit-card payment systems only, but *iKP* can be easily used to implement an electronic cheque system (e.g., the European *eurocheque* system). *iKP* does not support cash-like payment models (but *IBM* and *Europay* are jointly investigating the possibilities of an integration of an electronic purse technology and *iKP*).

5.1 Introduction

In early 1995 the *iKP* protocol was designed by security groups at IBM Research in Zurich and Yorktown. Technically many protocols that were proposed in 1994 and 1995 are very similar (e.g., *iKP, CyberCash, Netscape's Secure Courier*, *STT*, *SEPP, SET*), but *iKP* is the most systematic proposal, and it was the first where all design considerations were published. From the very beginning it was primarily thought as a contribution to standardisation rather than as a proprietary IBM technology.

iKP was designed in order

- to achieve a high level of integrity for all parties involved, thereby taking the different risks and policies into account;
- to provide privacy of payment data to the extend possible given that the existing credit-card model shall not be changed;
- to work with a minimum impact on the existing financial system;
- to avoid conflict with US export regulations and other national policies on usage of cryptography.

iKP supports what is required for secure payments, but

- it does not provide for negotiation of payment method, price, etc. — but contains a simple contract ("offer/order") procedure;
- it does not provide for untraceability of payments — but it protects the buyer's data from the seller, which is the best one can achieve given the requirement to build on top of the existing credit-card world;
- it does not provide means for secure delivery of information or fair exchange of payment for receipt — it provides receipts but does not enforce them.

iKP was first published in [BGHH 95], and in July 1995 submitted to IETF as an Internet draft [LiTs 95]. Based on this Internet draft *Mastercard's* proposal *SEPP* [SEPP 95] was designed and Version 1.1 published in September, Version 1.2 in November 1995. Thereafter *VISA* and *Mastercard* agreed on a common proposal, based on their experience with *SEPP* and *STT* [STT 96]. The result is called *SET* (Secure Electronic Transaction) and is still based on the design ideas and principles of *iKP*, although the details are quite different.

Based on [LiTs 95] and subsequent improvements and corrections, the IBM Zurich Research Lab is developing an *iKP*-prototype [vHST 96]. The following reviews the status of this prototype as of January 1996.

This prototype will be used in European trials in 1996 including the basic trials of *SEMPER* [Waid 96], although IBM has no plans to productize *iKP*. The committed timing of the trials makes it impossible to wait until standards like *SET* are consolidated without some limited deployment of *iKP* in the meantime.

5.2 Basic Settings

iKP involves three parties, buyer, seller, and the seller's acquirer, see Figure 4

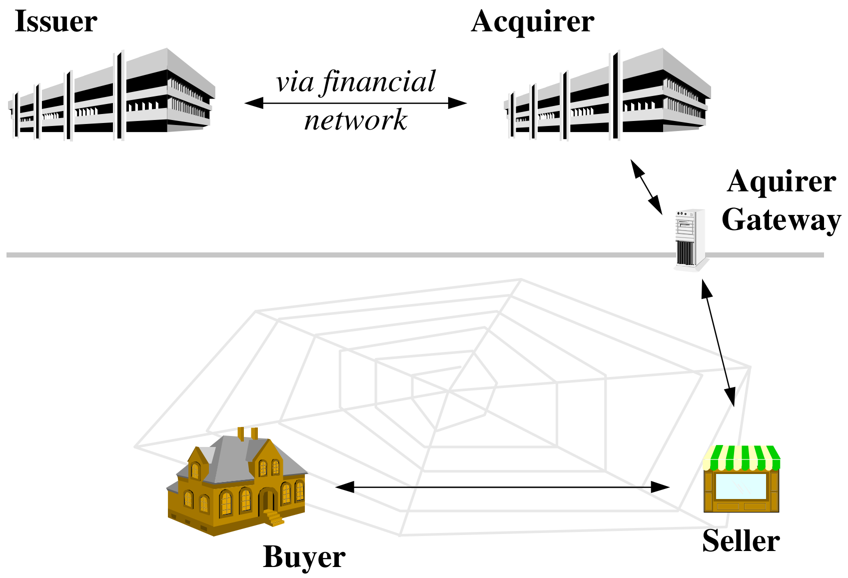

Figure 4 Basic Setting of *iKP*

Buyer and seller must have an already established relation with the financial system, i.e., the buyer must have a credit card and the seller must have a contract with an acquirer that allows it to accept the buyer's card. As a business requirement one should add that no additional contracts should be required in order to start electronic payments over the Internet.

The protocol employs the usual cryptographic primitives. For message authentication *iKP* uses

- digital signatures;

- plaintext-aware public-key encryption [BeRo 94] of secrets (like passphrases).

Plaintext-aware encryption is a mode of operation for a block encryption scheme that ensures message integrity, and is verifiably secure under some reasonable assumptions. In opposite to (properly initialised) digital signatures this method does not provide for non-repudiation.

For confidentiality, *iKP* uses two mechanisms:

- secret data that must be verified by the recipient but need not be transmitted since already known to the peer, are hidden in a *salted hash* (e.g., a number *N* would be hidden in *h(N, x)* where *x* is a random value known to sender and recipient);
- plaintext-aware public-key encryption.

iKP restricts encryption to those data that the acquirer must receive from the buyer in order to process the payment (e.g., credit card number, expiration date), and to those data that provide buyer authentication (like PINs or random salts used in hashes). The restriction is primarily done because of the US export rules, but does not reduce the security of *iKP* at all.[11]

The algorithms and key sizes actually used by the *iKP*-prototype are listed in Figure 5 (since *iKP* is implemented based on an abstract crypto-API, the algorithms can be replaced easily; e.g., instead of MD5 SHA might be used). It also includes the data for certification authorities, but certification is not specified by *iKP*.

	Hash	**Encryption**	**Signature**
Algorithm	MD5	RSA[12]	RSA
Certif. Auth.		none	1024 bit[13]
Acquirer		768 bit	1024 bit
Seller		none[14]	1024 bit
Buyer		none	1024 bit

Figure 5 Cryptographic algorithms and keys used in *iKP*

There are 3 basic variants of *iKP*: In 1KP, only the acquirer can digitally sign messages (i.e., there is only 1 party that has a public key, namely the acquirer). In 2KP, also the

11 In fact, if all parties can sign, and if the data used to identify the buyer are not usable for payments without the buyer's digital signature, one would not need any encryption. The reason why *iKP* uses encryption even if the buyer can sign is that, as a business requirement, on the Internet the same credit card data shall be used as in ordinary commerce. And currently the plain credit card data are sufficient to initiate mail order/telephone order (MOTO) transactions. The buyer is not liable for such transactions, but nevertheless forged transactions cause a lot of trouble for the buyer and for the merchant who accepted such a transaction.

12 *SET* uses the same key sizes, but uses DES key-envelopes for encryption (i.e, first a DES key is encrypted with RSA, and then the message with that DES key). In *iKP,* the data to be encrypted fit into a single block, and using DES would increase the implementation complexity and the political problems in countries like the USA or France.

13 *SET* specifies 2048 bit for the root CA. *iKP* uses a smaller modulus for implementation reasons only.

14 In *SET* the seller has an RSA key pair for encryption/decryption, with a modulus of lenght 768 bit. *iKP* does not support this. Non-payment data can be protected using protocols like *SSL* and *SHTTP*, and in *iKP* no payment data need to be encrypted towards the seller.

seller can sign (now there are 2 owners of public keys). And finally in 3KP, also the buyer can sign (3 owners of public keys).[15]

The *iKP*-prototype as well as *SET* [SET 96] support 2KP and 3KP only. The main reason is that 1KP does not allow effective seller authentication.

5.3 Message Flows

Figure 6 shows the message flows of 2KP and 3KP.

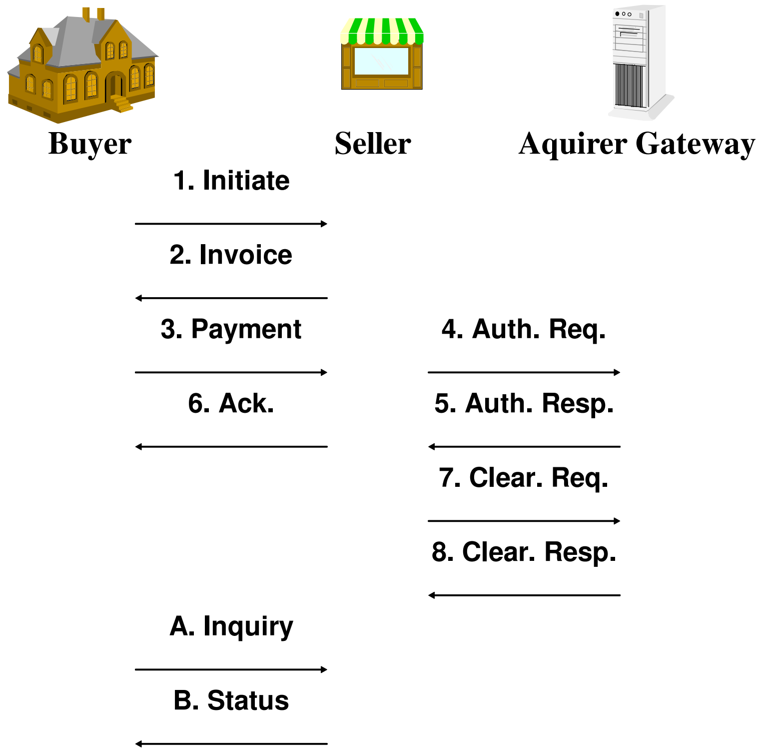

Figure 6 *iKP* message flows

Before the first flow, buyer and seller must have agreed on a common transaction context:

- The **amount** (incl. currency) to be paid. Within *iKP* the amount is never transmitted in the clear.
- A **description** of the transaction to which the payment will be linked, e.g., a description of goods or services, of external names of buyer and seller, of a delivery address, etc. This field is hidden from the acquirer by means of a salted hash (i.e., the acquirer sees hash(desc, salt_B) only).

The semantics of the *iKP* messages are almost obvious from their names:

1. **Initiate** contains the buyer's identity (which is a one-time identity for privacy reasons, computed as a salted hash of the buyer's account number), the identifier of

[15] Initially, *iKP* was an abbreviation for *i-key payment* protocol, with *i* = 1, 2, 3. But marketing required a reinterpretation of the name.

the brand name, and the transaction identifier used by the buyer (which is assumed to require no confidentiality protection).

2. **Invoice** is the seller's offer to the buyer. It contains the merchant's identity and transaction identifier, current date, an expiration date, and the acquirer's certificates.

   These data, the buyer's one-time identity from **initiate**, **amount** and a hash of **description** are those data buyer, seller, and acquirer have to agree on for the payment. Therefore the record of these data is called **common**. **Invoice** contains a hash of **common** computed by the seller which allows the buyer to check whether seller and buyer agree on them.[16]

   In 1KP, **invoice** would not be authenticated: It is easy to cause a lot of confusion and dissatisfaction by setting up a completely fake shop on the Internet, asking for payments. Of course, a non-existing seller could never clear these payments, i.e., there would be no direct financial benefit in this attack. Also the financial data like credit card numbers would not be revealed to the adversary, i.e., one cannot use the data received for attacks in the physical marketplace. 1KP is neither included in *SET* nor in the *iKP*-prototype.

   In 2KP and 3KP **invoice** is signed, and includes the seller's certificates. In this case **invoice** can be interpreted as an *offer* (as specified in **description**) that is valid for some time, and **payment** can be interpreted as an *order*, both together as a contract. But the legal interpretation of these digital signatures is out of scope of *iKP*.

   Initiate and **invoice** can be omitted in order to support email-order scenarios where buyer and seller are not communicating in real-time. In this case the seller's data required for constructing **payment** might come from a catalogue, CD-ROM, etc.

3. **Payment** contains the actual payment data corresponding to a paper credit-card slip (or cheque) in the physical world.

   Data that must not be revealed to the seller are encrypted with the acquirer's public key, namely *credit card number* and *expiration date,* the secret salt used in the construction of the one-time identity in **initiate**, and optionally authentication data for the buyer, e.g., a PIN or passphrase. Additionally it contains **amount**, a hash of **description**, and a hash of **common** computed by the buyer (which must be equal to the one computed by the seller in **invoice**, of course). This encrypted field is called **slip**.[17]

   In 3KP, **payment** is signed and includes the buyer's certificate. For privacy reasons this certificate includes the buyer's data (credit card number, expiration date) only as salted hash. The actual data and the salt are included in the encrypted **slip**[18].

   The seller is able to check the certificate and the signature, but not the contents of the encrypted **slip** which finally determine the validity of the payment.

---

16 *SET* uses a similar concept, called *dual signature*, which is adopted from *STT* [STT 95].

17 In *SET* seller's that need to know the exact payment data will receive the decrypted **slip** later from the acquirer gateway. The seller's certificate determines whether a seller can choose this option or not. *iKP* does not support this.

18 Note that two independently chosen salts are used for salting the buyer's identity in **initiate** and for salting the buyer's identity in the certificate.

In 2KP **payment** is authenticated only by means of the secret parts of the **slip**, provided it includes a PIN, and is therefore not disputable. See also Section 5.4.

4. **Authorisation-request** is the message by which the seller asks the acquirer to authorise the payment. Authorisation provides the seller a guarantee that the buyer's credit is sufficient. The actual transfer of money is initiated by **clearing-request**, but optionally, authorisation and clearing can be combined (in which case (6) and (7) are not present).

   Authorisation-request is a combination of **invoice** and **payment**. The acquirer can compile **common** after having decrypted **slip**, and can check whether all parties agree on **common** by comparing the hash values in **slip** (computed by the buyer), **invoice** (computed by the seller) and the value computed by the acquirer gateway itself. Implicitly this includes a check whether buyer and seller agree on the hash of **description**, i.e., on **description** itself, without revealing **description** to the acquirer.

   The acquirer gateway or the financial network is responsible for replay-detection of **authorisation-request.**

5. **Authorisation-response** is the acquirer's answer to **authorisation-request**. It is signed.

6. **Acknowledgement** is the seller's optional and signed receipt for the buyer[19]. A positive acknowledgement (**confirm**) is sent after successful authorisation. Optionally it can contain the acquirer's **authorisation-response** message. A negative acknowledgement (**cancel**) indicates that the payment cannot or will not be authorised/cleared. Note that *iKP* does not enforce an acknowledgement.

7. **Clearing-request** is the message by which the seller asks the acquirer to actually perform the payment, and refers to a previous **authorisation-response**. The amount to be cleared can be different from the amount authorized.

   Clearing-request is signed by the merchant.

   The acquirer gateway or the financial network is responsible for replay-detection.[20] The issuer is responsible for checking the cleared amount (in relation to the amount specified by the buyer's **slip** and the amount previously authorised).

8. **Clearing-response** is the acquirer's answer to **clearing-request**. It is signed.

A/B. **Inquiry/Status** are plain messages that allow the buyer to ask the seller about the status of the payment. In case **inquiry** is received after **acknowledgement** was sent, **acknowledgement** is repeated.

For a complete description see [vHST 96].

---

19 This is done indirectly: the signed **invoice** includes hashes of a secret values V_{accept} and V_{cancel}., and in **confirm** and **cancel** V_{accept} or V_{cancel} , resp., is given to the buyer as a "signature."

20 Actually "replays" are used to split the clearing of a single payment into parts. For instance, if not all goods are available, the merchant might create a **clearing-request** only for a part of the amount now, and might create a **clearing-request** for the full amount later, as soon as the goods are shipped.

5.4 2KP vs. 3KP

The differences between 2KP and 3KP are obvious:

- In 3KP the buyer is responsible only for payment orders that are signed by the buyer, which is supposed to be impossible for anybody else, including the payment system.
- In 2KP *with* secret passphrase in **slip**, the acquirer can easily forge payment orders, and there is no way to prove this. It is still secure against outsiders, even for relatively short passphrases (like PINs) since dictionary attacks are not possible.
- In 2KP *without* secret anybody who knows the payment data can forge payments (which is still slightly better than in today's mail-order/telephone orders since the seller does not receive the payment data).

SEPP and *SET* specify 2KP and 3KP. For several reasons the initial *SET* systems might use 2KP only (with or without secret PINs):

- By defining the business rules appropriately (like in MOTO transactions) non-repudiable proofs of the buyer's authorisation are not required. (There is, of course, always the problem that acquirers and issuers do not necessarily handle actual disputes in favour of the buyers, even if they are not liable according to the rules.)
- The costs for certifying all buyers are extremely high, and building a global public-key infrastructure of this size is not well understood yet.
- The risk of compromise of a secret signature key or of the signature program stored on an insecure PC of a buyer is considered as too high. This possibility would almost destroy the legal binding of digital signatures, i.e., for 3KP one needs secure wallets for the buyer to guarantee true security.

5.5 Certificates

iKP assumes a certificate infrastructure but does not dictate which one. The *iKP*-prototype has an interface to a certification manager, i.e., it is independent of how exactly certificates look like and are managed. The actual certification management system used in the prototype is a non-standard solution that is designed to work in a trial environment.

5.6 Security

5.6.1 Integrity

Figure 7 summarises the integrity properties of 3KP and 2KP with shared secret ($2KP^+$) and 2KP without shared secret ($2KP^-$). An entry in Row *X*, Column *Y* describes how the authorisation by Party *Y* is proven towards Party *X*. A proof of authorisation is called "disputable" if a third party like a court can verify that proof, i.e., if it provides non-repudiation.

	Bank	**Buyer**	**Seller**
Bank		**1. Payment** **2KP$^-$: No proof,** but sanity checking. **2KP$^+$: Non-disputable proof** via encrypted shared secret (PIN) in **slip**. **3KP**: **Disputable proof** via signed **slip.** Encryption is done to protect credit card data only.	**2. Authorisation/ Clearing Request** **Disputable proof:** **auth-request** / **clear-request** are signed by seller.
Buyer	**3. Notification** Out-band notification in credit card statement. Note that this is not part of *iKP*, but required for final authorisation in **2KP$^-$**!		**4. Invoice / Confirm** **Disputable proof, but not enforced:** **invoice** / **ack.** are signed by seller. There is no mechanism to enforce **ack**.
Seller	**7. Authorisation/ Clearing Response** **Disputable proof:** All **responses** are signed by acquirer.	**8. Payment** **Not required.** In 3KP, seller can verify signature in **payment** which can be interpreted as part of a contract.	

Figure 7 Integrity Properties of *iKP*: Each entry shows how a payment is authorized *by* the party listed in the top row *towards* the party listed in the left column. The entry title shows which message transports the authorisation.

The essential relations in Figure 7 are (1) and (7). In case of 2KP$^-$ Relation (3) is also essential since the buyer cannot securely authorise the payment in (1), i.e., they must actively deny any unauthorized payment in (3).

Relation (2) is nice to have since it prevents simple attacks that would cause a lot of confusion, but it is not essential.

The "invoice" part of Relation (4) is considered essential as well. Without the signed **invoice** the buyer cannot know whether they are paying to the intended recipient, or, e.g., to someone who has replaced the seller's data. Similarly there would be no disputable link between the payment and **description**. It is arguable whether this is a natural part of a payment system, or whether it should be rather part of the negotiations that must happen before a payment. But it is important that it is done somewhere. This seller authentication and implicit contract signing procedure is the main reason why *iKP* considers the seller's signatures as mandatory.

In 3KP, Relation (8) enables sellers to verify that the payment is initiated by an existing buyer. For small amounts this might be a sufficient argument for the seller to perform a service at their own risk, i.e., without immediately sending **authorisation-request** to the acquirer.

5.6.2 Confidentiality

iKP ensures that acquirer and seller receive only those buyer data they actually need in order to perform the required credit card transaction. The acquirer receives all payment data, but not **description**. The seller does not receive the buyer's credit card details (in contrast to the situation in conventional credit card payments)[21].

In 2KP payments are not linkable by sellers (**invoice** includes a one-time identity only).[22] In 3KP the buyer's certificate is visible to the seller, and the same for all payments, i.e., here payments are linkable. Since the seller does not need to check the buyer's signature one could prevent this by encrypting the certificate under the acquirer's public key, but *iKP* does not support this.

iKP does not provide any anonymity against the payment system.

5.7 Status of the *iKP*-Prototype

The *iKP*-prototype will be ready by early 1996. It will be used in several European trials. The architecture is shown in Figure 1. The *Payment API* is independent of *iKP* and could be used also for later standard-based products.

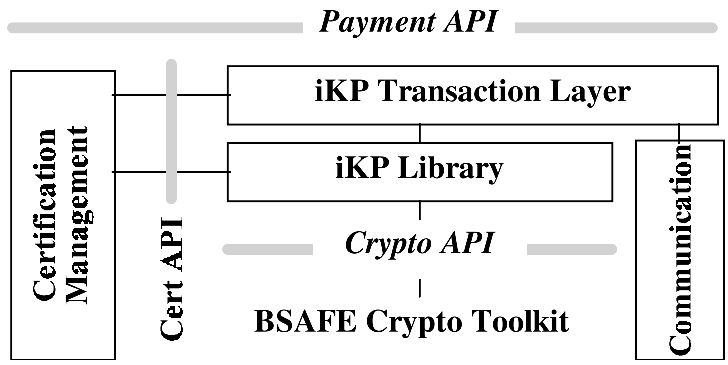

Figure 8 Architecture of the *iKP*-Prototype

The *iKP*-prototype consists of two main components:

- The *iKP Library* is a relatively short C-program that evaluates and/or computes all *iKP* messages. All required cryptographic operations are performed by RSA's *BSAFE* toolkit. The crypto API is specific for *iKP* (which would allow to replace BSAFE by another crypto library if necessary).
- The *iKP Transaction Layer* (written in C++) manages payment transactions, i.e., keeps state, retransmits messages if necessary, etc.[23]

The *communication module* is responsible for sending and receiving the *iKP* messages (the prototype will support communication via *http* and *mail*). The *certification mod-*

[21] In *SET* a seller might receive these details, see FN 17.

[22] Of course, in most cases the seller will be able to link payments by looking at the *context,* like the buyer's address. In order to prevent this linkability one would have to use techniques like *anonymous remailers* or *mixes* [Chau 81, GüTs 95].

[23] The prototype does not use a standard transfer syntax or encoding for messages (unlike *SET* which uses ASN.1).

ule handles the user's certificates (the prototype will support only a non-standard certificate format, i.e., not *X509*).

The prototype is being implemented for *AIX* and *Windows 3.1*, and "exportable," i.e., IBM received an export permit from the US government.

5.8 Extensions of *iKP*

An extension of *iKP* for micropayments is described in [HaSW 95]. Basically it is an application of the scheme used by *CAFE* for phone ticks [Pede 95]. The idea is to replace the amount in **payment** by a field containing $(n, f^n(x), u)$, where n is a number, f is a fixed one-way function, x is a value randomly and secretly chosen for that payment, and a u is an amount. Micropayments are done by revealing successively $f^{n-1}(x), f^{n-2}(x)$, ..., x, whereby each value is worth 1 unit of money as specified by u. The *iKP*-prototype does not include this extension.

6 Standardisation

The European Standardisation Organisation, *CEN*, as well as *Europay, MasterCard, VISA* (for that purpose known as *EMV*) are working on standards for smartcard-based electronic payment systems. While the *CEN* standard on an Intersector Electronic Purse already exists, the current *EMV* specifications [EMV 95] do not include electronic purse functions. No standardisation efforts towards an untraceable offline payment system exist.

Two initially competing proposals, *STT* and *SEPP*, for standards for credit-card payment schemes were published by *VISA* [STT 95] and *Mastercard* [SEPP 95], respectively, and recently merged into a common proposal, called *SET* — Secure Electronic Transaction [SET 96]. First prototypes of *SET* toolkits already exist, and products are expected to be available before the end of 1996. *SET* was designed by GTE, IBM, Microsoft, Netscape, SAIC, Terisa and Verisign, and it is based on the same design ideas and principles as *iKP* (like *STT* and *SEPP*). It is not known yet whether all other credit card companies will adopt *SET* as their payment system.

During the last months several proposals for credit-card payment systems were submitted to the *Internet Engineering Task Force* (*IETF*), namely *FirstVirtual, CyberCash,* and *iKP*, and a working group on electronic payment systems was established in December 1995. But as a result of joint statements by *MasterCard* and *VISA*, the IETF working group works on negotiation and encoding (transport) of payment protocols only, rather than on the protocols themselves. A first step in this direction was [EaBo 95]. *CommerceNet* and the *W3 Consortium* started a joint project supporting this IETF work.

Currently, the discussions around *SET* dominate the stage of *Internet* payment systems. But there is a parallel demand for international standards on electronic cash-like payment schemes and schemes for micropayments.

7 Summary

This high-level overview should have made clear that the technology that is necessary for secure electronic Internet payment systems already exists. Achieving security for all parties, inclusive perfect untraceability for the buyer, is possible.

The question “Which payment system will be used on the Internet?” will not have a single answer. Several payment systems will coexist:

- Micro-payments (less than 1 ECU), low-value payments (1-100 ECU) and high-value payments have significantly different security and cost requirements.

  Possibly, high values will be transferred using non-anonymous, on-line payment systems based on asymmetric cryptography, implementing a cheque-like or credit-card-like (e.g., *SET*) payment model. As soon as smartcard readers are available at PCs and workstations, small amounts might be paid using pre-paid off-line payment systems that provide a certain degree of untraceability (like real cash).

- Payment systems with and without tamper-resistant hardware at the buyer will co-exist for some time. Ultimately, payment systems based on smartcards and electronic wallets (having secure access to some display and keyboard, and communicating with the buyer’s terminal via an infrared interface) will become dominant, since they clearly provide better security and enable the buyer to use untrusted terminals without endangering security.

- Maybe a few almost equivalent payment systems will coexist for the same areas of application (i.e., payment model and maximum amounts). The reasons are various “cultural” differences in the business and payment processes (e.g., between the U.S. and Europe), national security considerations that might disable some solutions in some countries, and competition between payment system providers.

8 References

For a collection of WWW pointers see
<http://www.zurich.ibm.ch/Technology/Security/sirene/outsideworld/ecommerce.html>.

Ande 94 — *R. Anderson:* **Why Cryptosystems Fails;** Communications of the ACM 37/11 (1994) 32-41. <ftp://ftp.cl.cam.ac.uk/users/rja14/wcf.ps.Z>

APAW 96 — *J. Abad Peiro, N. Asokan, M. Waidner:* **Payment Manager — Overview;** IBM Zurich Research Lab, 21 Mar 1996; *SEMPER* Activity Paper 212ZR054. <http://semper.zurich.ibm.com/info/>

BBCM 94 — *J.-P. Boly, A. Bosselaers, R. Cramer, R. Michelsen, S. Mjølsnes, F. Muller, T. Pedersen, B. Pfitzmann, P. de Rooij, B. Schoenmakers, M. Schunter, L. Vallée, M. Waidner:* **The ESPRIT Project CAFE - High Security Digital Payment Systems;** ESORICS '94, LNCS 875, Springer-Verlag, Berlin 1994, 217-230. <http://www.zurich.ibm.com/Technology/Security/sirene/publ/BBCM1_94CafeEsorics.ps.gz>

BeMe 92 — *S. Bellovin, M. Merritt:* **Encrypted Key Exchange: Password-based Protocols Secure Against Dictionary Attacks;** IEEE Symposium on Research in Security and Privacy, May 1992.

BeRo 94 — *M. Bellare, P. Rogaway:* **Optimal Asymmetric Encryption;** Pre-proceeding of Eurocrypt '94, May 9-12, 1994, University of Perugia, Italy 103-113.

BGHH 95 *M. Bellare, J. A. Garay, R. Hauser, A. Herzberg, H. Krawczyk, M. Steiner, G. Tsudik, M. Waidner:* **iKP — A Family of Secure Electronic Payment Protocols;** Usenix 1995. <http://www.zurich.ibm.ch/Technology/Security/publications/1995/ikp.ps.gz>

BLSS 95 *J. Benaloh, B. Lampson, D. Simon, T. Spies, B. Yee:* **The Private Communication Technology Protocol;** Microsoft, Internet Draft, October 1995.

Bore 95 *N. Borenstein et.al. (First Virtual):* **Perils and Pitfalls of Practical CyberCommerce - The Lessons of First Virtual's First Year;** Frontiers in Electronic Commerce, Austin, Texas, October 1995.

Bran 94 *S. Brands:* **Untraceable Off-line Cash in Wallet with Observers;** Crypto '93, LNCS 773, Springer-Verlag, Berlin 1994,302-318.

BüPf 89 *H. Bürk, A. Pfitzmann:* **Digital Payment Systems Enabling Security and Unobservability;** Computers & Security 8/5 (1989) 399-416. <http://www.zurich.ibm.com/Technology/Security/sirene/publ/BuePf_89.ps.gz>

CaPS 94 *J. L. Camenisch, J.-M. Piveteau, M. A. Stadler:* **An Efficient Electronic Payment System Protecting Privacy;** ESORICS 94, Brighton, November 1994.

Chau 81 *D. Chaum:* **Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms;** Communications of the ACM 24/2 (1981) 84-88.

Chau 83 *D. Chaum:* **Blind Signature System;** Crypto '83, Plenum Press, New York 1984, 153.

Chau 89 *D. Chaum:* **Privacy Protected Payments — Unconditional Payer and/or Payee Untraceability;** SMART CARD 2000, North-Holland, Amsterdam 1989, 69-93.

Duka 92 *S. Dukach:* **SNPP: A Simple Network Payment Protocol;** Computer Security Applications Conference,1992.

East 95 *D. E. Eastlake:* **An Application Level Internet Payment Syntax;** CyberCash, Internet Draft, July 1995.

EaBo 95 *D. E. Eastlake, B. Boesch:* **Universal Payment Preamble;** CyberCash, Internet Draft, November 1995.

ECBY 95 *D. E. Eastlake, B. Boesch, S. Crocker, M. Yesil:* **CyberCash Credit Card Protocol Version 0.8;** CyberCash, Internet Draft, October 1995.

EMV 95 **Integrated Circuit Card Specifications for Payment Systems;** Europay/MasterCard/VISA, October 1994/June 1995.

EvGY 83 *S. Even, O. Goldreich, Y. Yacobi:* **Electronic wallet;** Crypto '83, Plenum Press, New York 1984, 383-386.

GSPT 95 *D. K. Gifford, L. C. Stewart, A. C. Payne, G. W. Treese:* **Payment Switches for Open Networks;** IEEE COMPCON, March 95.

GüTs 96 *C. Gülcü, G. Tsudik:* **Mixing Email with BABEL;** 1996 Symp. on Network and Distributed System Security, Feb. 1996, San Diego, California. <http://www/Technology/Security/publications/1995/gt95.ps>

GuUQ 92 *L. Guillou, M. Ugon, J.-J. Quisquater:* **The Smart Card: A Standardized Security Device Dedicated to Public Cryptology;** in: Gustavus J. Simmons: Contemporary Cryptology – The Science of Information Integrity; IEEE Press, Hoes Lane 1992, 561-613.

HaSW 95 *R. Hauser, M. Steiner, M. Waidner:* **Micro-Payments based on iKP;** IBM Zurich Research, October 1995. IBM Research Report RZ 2791; to be presented at SECURICOM 96, Paris 1996. <http://www.zurich.ibm.ch/Technology/Security/publications/1996/HSW96.ps.gz>

HiEl 95 *K. Hickman, Taher ElGamal:* **The SSL Protocol;** Netscape, Internet Draft, June 1995.

JaWa 95 *P. Janson, M. Waidner:* **Electronic Payment over Open Networks;** SI Informatik / Informatique 3 (1995) 10-15. <http://www/Technology/Security/publications/1995/JaWa95.dir/JaWa95e.html>

LiTs 95 *M. Linehan, G. Tsudik:* **Internet Keyed Payment Protocol (iKP);** IBM, Internet Draft, July 1995. <http://www.zurich.ibm.com/Technology/Security/extern/ecommerce/draft-tsudik-ikp-00.txt>

LoMP 94 *S. H. Low, N. F. Maxemchuk, S. Paul:* **Anonymous Credit Cards;** 2nd ACM Conference on Computer and Communication Security, Fairfax 1994.

Long 83 *L. Longpre:* **The use of public-key cryptography for signing checks;** Crypto 82, Plenum Press, New York 1983, 187-197.

LoMP 94 *S. H. Low, N. F. Maxemchuk, S. Paul:* **Anonymous Credit Cards;** 2nd ACM Conference on Computer and Communication Security, Fairfax 1994.

Mana 95 *M. Manassee:* **Design considerations for lightweight payment protocols.** Usenix 95.

MeNe 93 *G. Medvinsky, C. Neumann:* **NetCash: A design for practical electronic currency on the Internet;** 1st ACM Conference on Computer and Communications Security, acm Press, New York 1993, 102-106.

NeMe 95 *C. Neuman, G. Medvinsky:* **Requirements for Network Payment: The NetCheque™ Perspective;** IEEE COMPCON, March 95.

Neum 95 *C. Neuman:* **Security, Payment, and Privacy for Network Commerce;** IEEE Journal on Selected Areas of Communications 13/8 (1995) 1523-1531.

Pede 95 *T. Pedersen:* **Electronic payments of small amounts;** Technical Report, Aarhus University, Computer Science Department, August 1995.

Pfit 84 *A. Pfitzmann:* **A switched/broadcast ISDN to decrease user observability;** 1984 International Zurich Seminar on Digital Communications, etc., IEEE 1994, 183-190.

PPSW 95 *A. Pfitzmann, B. Pfitzmann, M. Schunter, M. Waidner:* **Mobile User Devices and Security Modules: Design for Trustworthiness;** IBM Research Report RZ 2784, 1996. <http://www.zurich.ibm.ch/Technology/Security/publications/1996/PPSW96.ps.gz>

PWP 90 *B. Pfitzmann, M. Waidner, A. Pfitzmann:* **Rechtssicherheit trotz Anonymität in offenen digitalen Systemen;** Datenschutz und Datensicherung DuD 14/5-6 (1990) 243-253, 305-315. <http://www.zurich.ibm.ch/Technology/Security/sirene/publ/PWP_90anonyZsys.ps.gz>

ReSc 95 *E. Rescorla, A. Schiffman:* **The Secure HyperText Transfer Protocol;** Enterprise Integration Technologies, Internet Draft, July 1995.<http://www.commerce.net/cgi-bin/textit?/information/standards/drafts/draft-ietf-wts-shttp-00.txt>

RiSh 95 *R. Rivest, A. Shamir:* **PayWord and MicroMint;** MIT / Weizmann November 1995.

Rolf 94 *R. Rolfe:* **Here Comes Electronic Cash;** Credit Card Management Europe, January/February 1994, 16-20.

Schn 95 *B. Schneier:* **Applied Cryptography;** John Wiley & Sons, 1994, 1995.

Seel 95 *G. Seel:* **Electronic Payments Design Considerations;** FSTC Electronic Commerce Project 1995.

SEPP 95 *Cybercash, GTE, IBM, MasterCard, Netscape:* **Secure Electronic Payment Protocol;** MasterCard, November 3, 1995, Version 1.2.

SET 96 *Mastercard, VISA:* **Secure Electronic Transactions;** Draft, February 23, 1996. <http://www.mastercard.com/set/set.htm>

SiTy 95 *M. Sirbu, J. D. Tygar:* **NetBill: An Internet Commerce System;** IEEE COMPCON, March 95.

SSBR 95 *L. H. Stein, E. A. Stefferud, N. S. Borenstein, M. T. Rose:* **The Green Commerce Model;** First Virtual Holding Inc., Internet Draft, May 1995.

STT 95 *VISA, Microsoft:* **Secure Transaction Technology Specifications;** VISA International, September 26, 1995, Version 1.0.

vHST 96 *E. van Herreweghen, M. Steiner, G. Tsudik:* **Zurich iKP Prototype (ZIP);** IBM Zurich Research; 1996. <http://www/Technology/Security/extern/ecommerce/iKP_references.html>

Waid 96 *M. Waidner:* **Development of a Secure Electronic Marketplace for Europe;** IBM Zurich Research Lab, 19 Feb 1996; *SEMPER* Activity Paper 431ZR056. <http://semper.zurich.ibm.com/info/ >