Secure PC-Franking for Everyone

Gerrit Bleumer

Francotyp-Postalia, D-16547 Birkenwerder, Germany

Abstract. PC franking systems allow customers to download postage value into their PCs and to print postage value onto envelopes or mailing labels by using regular desktop printers connected to their PCs. In emerging standards, such as the IBIP program of the US postal services, the resulting imprints are called *indicia*, and they are printed as 2-D bar codes that are easy and reliable to scan while passing through the mail delivery system. In contrast to the IBIP program, which requires a tamper responsive postal security device for each customer, we propose a pure software solution for PC franking systems that is more secure for the postal services, gives more privacy to the customers and has the potential of being much cheaper than any IBIP compliant franking system. In contrast to existing PC-franking products that require no hardware postal security device, our solution works off-line. Customers are not required to be online while printing indicia. Our solution operates an existing offline e-cash scheme over certain elliptic curves rather than over finite fields. [1]

Keywords: Open franking systems, offline electronic cash, blind signature, elliptic curves.

1 Introduction

Many companies are moving towards electronic commerce and so do the postal services in many countries around the world. Traditionally, larger companies and organizations use postage meter machines for franking mail. These meter machines are licensed to identified individuals and require some dedicated connection to the postal service in order to be charged with postage (*closed franking systems*). For instance, mechanical meter machines are charged via physical tokens. More recent electronic meter machines are charged by using leased lines or telephone lines to download postage off of a postage server. Furthermore, postage meter machines are sold or leased to registered customers only, and the machines are inspected by the postal services on a regular basis. There are about 1.7 million meter machines installed in the US and they comprise about 30% of the US mail volume. In 1998, postage meters accounted for more than 21 of 60 billion dollar in revenue of the USPS. At the same time, the USPS suffers about

[1] EC-Web'00), LNCS 1875, pp. 94-109, 2000.

a 100 million dollar loss per year from meter fraud. The quest is—not only in the US—for more secure and more cost efficient franking systems.

While more and more small offices and home offices (soho) have enough computing and printing power sitting right on their desktops (more than 40% of US households) and Internet access is cheap, the time has come for franking systems that allow to download postage via open networks such as the Internet (*open franking systems*) and do not require dedicated hardware and regular inspections. A PC can be used to download postage and a standard printer can be used to print out postage onto regular paper envelopes, labels, etc. The resulting printouts are called *indicia*. These are scanned and verified when the respective mail pieces pass the postal sorting centers. The US postal service is one of the first who has specified open franking systems in their Information Based Indicia Program [IBIP99o]. Commercial products complying to the IBIP standard are appearing on the US market, e.g., e-stamp (www.e-stamp.com), stamps.com, etc. Open franking systems should be more robust against fraud than closed systems because users of open franking systems are not registered and their equipment is not undergoing regular postal inspections. However, open franking systems must be substantially cheaper than closed franking systems because otherwise they will not enter the mass market. We call this the *low-cost-high-security dilemma* of open franking systems.

Our proposal features (i) a pure software solution, (ii) offline franking in the sense that the sender can produce and print indicia without having an online connection to any other participant such as a trusted third party, (iii) security as strong as state-of-the-art cryptographic payment systems and (iv) strong customer privacy protection. Such a software solution has the potential of being cheaper than today's closed franking systems and also cheaper and more privacy protecting than any PC franking system complying to the IBIP program. That is because the IBIP program requires each customer to use a tamper responsive hardware security module to hold the signature keys necessary to produce customer specific indicia. Our solution can save the additional cost of producing such hardware security modules and yields more customer privacy because it avoids customer specific indicia. Obviously this escapes the low-cost-high-security dilemma.

We first give a brief overview of open franking systems before we identify the relevant security requirements (Section 2). The new ideas of this work are described in Section 3. All notation used throughout is introduced in Section 4. The details of the new proposal are presented in Section 5. A comparison with the open franking system specification according to the IBIP program is given in Section 7. Design and Performance Options are discussed in Section 6. Finally, the results are summarized in Section 8.

2 Security Requirements on Franking Systems

Before discussing the security requirements, we run through a typical franking cycle. The postal service in each country acts in two simultaneous capacities. The

postal accounting system (P) maintains a special currency, stamps namely, and the mail delivery system (M) collects, sorts and distributes mail and invalidates the respective stamps in return. For electronic franking, P provides a postage server (usually through some third party) from which a sender (S) can download postage electronically whenever she likes. The postage can be stored on a PC harddrive, a chipcard or any other persistent storage medium. Typically, senders are charged for the postage at download time. Frequent senders may use postal accounts, occasional senders may pay by credit card, debit card, wire, check or electronic cash. Regular letter envelopes or postcards can be printed directly, and labels can be used for larger mail pieces. Printed indicia must at least contain their face value and an electronic signature to authorize this value. In addition, they may support certain functions of the mail delivery system including fraud prevention. For example, they may contain the delivery address, an expiration date, etc. From a privacy perspective, the sender's identity should not be included, unless the sender wishes so. The indicium contains a human readable portion and a machine-only readable portion, which is typically displayed using a two-dimensional bar code such as Data Matrix or PDF417 (see section 6.3). Once it is printed and the piece of mail is sent, it passes through the mail delivery system M where the indicia are scanned and invalidated.[2] If an indicium shows sufficient face value, the mail piece is delivered to the recipient (Fig. 1). The

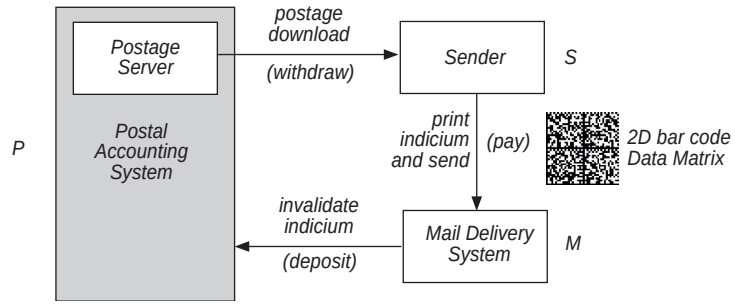

Fig. 1. Franking System

postal services require a franking system to counter fraud as much as possible. Customers require that their postal accounts are secure against false access, and they usually like the option of sending mail anonymously, i.e., without disclosing personal data. These security requirements are listed in more detail below:

Unforgeability: At any point in time the mail delivery system shall not deliver more mail than what has been paid for (since system startup). Important sub-requirements of unforgeability are the following.

[2] For performance reasons, the mail sorting centers might verify only a small percentage of indicia.

Double Spending Prevention: After a sender has downloaded x $ of postage, she shall not be able to print out indicia whose overall value exceeds x. In open franking systems the recipient's address (e.g. in the form of a 9-digit ZIP code) and a time stamp are usually included in the indicium. This way, re-using a previous indicium allows little fraud, even without further cryptographic protection. In closed franking systems, the franking process is more separate from the mail addressing process, so including the recipient in the indicia is not an option. In this case, Xerox copies of indicia can be detected (and thereby discouraged) if the mail delivery system maintains a database of indicia which have been processed and delivered. Whenever an indicium turns up a second time it can be rejected. Another way of countering Xerox copies is by using red fluorescent ink, which is hard to reproduce with regular Xerox equipment.

Double Spender Detection: After a sender has used the same piece of postage more than once, he can be identified, for example by means of the postal account from which the piece of postage was downloaded.

Unframeability: This requirement makes sense for schemes with double spender detection only. It means that the postal services cannot wrongfully accuse anyone of double spending.

Indicia unlinkability/sender anonymity: Any two indicia shall not reveal whether they originate from the same sender, unless the sender wishes so. This requirement implies the weaker requirement of sender anonymity, which means that indicia do not reveal their sender's identity.

For general purpose payment systems it has been argued that unconditional unlinkability of payments and therefore unconditional payer anonymity is too strong in practice, because it encourages money laundering, etc. [SN92]. For electronic postage, however, a strong privacy requirement like indicia unlinkability is an interesting option. There is minimal risk of misuse because postage values are relatively small, and indicia will hardly attract big money. In addition there are some convenience requirements:

Divisibility: After downloading an amount x of postage, a sender should be able to produce any set of indicia which together represent no more postage value than x. This escapes the annoying situation with conventional stamps where, for example, a sender cannot produce the value 60c if he has only a number of 33c stamps.

Online Printing: There is no separate download transaction, but each time senders wants to print indicia, they connect to a postal server in order to compute and print a proper indicium. This approach is taken for example by Pastor's CRYPTOPOST system [P91] or the more recent PC franking solutions of stamps.com.

Offline Printing: The sender downloads postage in advance and is then able to produce and print indicia without having an online connection to any postal server or other trusted third party. Offline printing is probably more convenient for senders.

3 What's New

In order to obtain an efficient cryptographic solution for open franking systems, we consider prepay e-cash schemes and map the principals of a franking system as follows: The sender takes the role of a *payer* and the mail delivery system M takes the role of a *merchant*. The postal accounting system P takes the role of the *payer's bank* and of the *merchant's bank* at the same time. Downloading postage off the postage server corresponds to withdrawing money from the sender's account. Printing an indicium and putting the corresponding piece of mail into a mailbox corresponds to buying a service and paying for it. Finally, the mail delivery system invalidates the indicium. This roughly corresponds to a merchant depositing coins into his bank, i.e., the postal accounting system. Since the mail delivery system is a centrally run organization, it can be viewed as one huge merchant.

One of the critical issues in e-cash schemes is double spending, i.e., payers who try to pay more than once with the same coin. In e-cash schemes this can be prevented (*double spending prevention*) by requiring a trusted third party to be online during the payment transaction (*online e-cash*), or by requiring some trusted tamper resistant hardware in the payer's hand that overlooks the payment transaction (*offline e-cash*). Should double spending occur nevertheless, then the double spender can be efficiently identified with both types of systems (*double spender detection*) [B93,BC97].

In open franking systems double spending prevention seems not by all means necessary, because the incentive for double spending can be brought close to zero by non-cryptographic means. In fact, the mail delivery system may require a customized indicia currency where a different type of indicia is required for each recipient each day or each week.[3] This way, a fraudster who has sent a letter to recipient R could re-use that indicia only for a second letter sent to the same recipient R on the same day or week. Only in this rare case is re-using not detectable. However, re-using can be detected if the second indicium is compared to all previous indicia scanned by the postal services, which is possible in $\log n$ time, where n is the number of all indicia previously scanned. The double spending detection mechanism can reveal the fraudster's identity, e.g., his postal account, and appropriate measures can be taken. For example, the fraudster could be charged a penalty fee and if he pays in time, his mail piece could be delivered to the specified recipient. This specialized indicia currency can be used in franking systems, because the postal accounting service knows all the specifics, i.e., recipient, time of mailing, etc., anyway and can use them for verifying the indicia. Thus we can meet the DOUBLE SPENDING PREVENTION requirement by properly customized indicia and the DOUBLE SPENDER DETECTION and the OFFLINE PRINTING requirements by employing an offline e-cash scheme.

[3] If this appears too weak against organized groups who are geographically distributed, the sender's mailing area could be used in addition to recipient address and time of mailing.

In order to achieve INDICIA UNLINKABILITY we use ideas from an untraceable e-cash scheme by Brands [B93,B94,B96], where payment transactions cannot be linked to corresponding withdrawal transactions. Thus, the postal service cannot figure out which indicium corresponds to which postage download nor whether two indicia originate from the same sender.

4 Notation and Definition

Since all the following protocols are based on the intractability of computing discrete logarithms, the following definitions are useful. Let q be prime. Let G_q be a family of finite, multiplicative, Abelian groups of order q. Furthermore, let integral powers (g^x with $g \in G_q$ and $x \in \mathbb{Z}$) be defined by iterated group multiplication. Given a generator $g \in G_q$ and a randomly chosen element $z \in G_q$, the smallest non-negative integer x that satisfies $z = g^x$ is called the *discrete logarithm* of z with respect to g, if such x exists. More generally, if $g_1, \ldots, g_l$ are generators of G_q, then the tuple $(x_1, \ldots, x_l)$ is called a *discrete representation* of z with respect to $g_1, \ldots, g_l$ if $z = \prod_{i=1}^{l} g_i^{x_i}$ holds.

In the following we consider families of groups G_q that have efficient algorithms for multiplying, choosing group elements uniformly at random and testing equality, and in which computing discrete logarithms and discrete representations is hard, i.e., computing discrete logarithms and discrete representations is not polynomial-time in the bitlength of q. Yet no candidates have been proven to exist with respect to the latter requirement, but there are candidates for which many believe that that the *discrete logarithm assumption* DLA and the (*discrete representation assumption* DRA hold. Both assumptions are equivalent [CHP91,B93].

One candidate for the family G_q are the large cyclic subgroups of the multiplicative groups $\mathbb{Z}_p^*$ of residues modulo a large prime p [S91]. Other candidates are suitable elliptic curves over finite fields [MOV97,IEEE]; more precisely large subgroups of prime order q of suitable elliptic curves. [4] It is generally believed that computing discrete logarithms over a cyclic subgroup of residues modulo a prime p of length 1024 bit is approximately as hard as computing "discrete logarithms" over a suitable elliptic curve on an order the size of 160 bit [O99]. (We use quotation marks, because by convention, elliptic curves are written additively.) In the following, elliptic curve notation is obtained by replacing products and powers in G_q by curve addition and scalar multiplication.

We denote protocols and algorithms by a declaration and a definition. A *protocol declaration* consists of (i) the formal output parameters, followed by (ii) an assignment arrow, followed by (iii) the protocol name and (iv) the formal input parameters in parenthesis. To enhance readability, all input and output parameters of a participant are enclosed in square brackets labeled by the participants initial. Values of formal input parameters are called *private input* or

[4] Which elliptic curves are suitable is subject to ongoing research. By time of writing, no super-singular or anomalous elliptic curves, or curves of high genus are suitable because they have efficient algorithms to compute discrete logarithms [O99].

common input if these parameters are given to only one or to all participants of the protocol. A *protocol definition* is denoted in matrix form. Actions of each participant are listed in columns, and each column is labeled by its participants name. Consecutive actions are displayed in consecutively numbered rows, which are called the *steps* of the protocol.

Protocol actions are denoted by usual mathematical notation and a few special symbols. Choosing an element uniformly at random from a set A and assigning it to a variable a is denoted $a \in_R A$. Evaluating an expression E and assigning the result to a is denoted by a left arrow $a \leftarrow E$. By $\mathcal{H}$ we denote a pseudo-random hash function [FS87] that, on input any binary string, returns a value in $\mathbb{Z}_q$. We relax the notation by allowing any number of arguments to $\mathcal{H}$ meaning that their binary representations are concatenated and then fed to $\mathcal{H}$. Arithmetic operations are either in G_q, i.e., multiplication mod p or in $\mathbb{Z}_q$, i.e., addition and multiplication mod q. We omit the "(mod p)" and "(mod q)" whenever the modulus is clear from the context. Transmitting the value of a variable a from participant Alice to participant Bob is simply denoted by a labeled arrow $\xrightarrow{a}$[5] that stretches from Alice's to Bob's column. Calls of protocols or algorithms are denoted as usual, i.e., by instantiating their formal parameters with respective actual parameters. The phrase "proceed iff P" with P a Boolean predicate indicates that the protocol execution proceeds if and only if P holds. Otherwise, the protocol is aborted and the participants return a corresponding exception.

5 A Software Solution

Let g_1, g_2, G, G_0 be four generators of G_q, which are chosen independently and uniformly at random at system startup time. The postage server chooses a private key $x \in \mathbb{Z}_q^*$ uniformly at random and computes the corresponding public key $y = G^x \bmod p$. The values G_q, g_1, g_2, G, G_0, y are publicly available.

The following proposal employs two types of currency. *Pieces of postage*, which are downloaded off the postage server, and *indicia*, which are printed onto the mail. Indicia occur in digital and printed form. The printed form is an easy to scan graphical encoding of the digital form, e.g., a two-dimensional bar code. Each piece of postage and each indicium has a monetary value, but indicia constitute a more specialized currency in that they contain additional information about the mailing recipient, the date and time of sending and possibly also about the origin from where the indicium is sent.

Pieces of postage PoP are tuples (A, B, σ), where A, B are elements of G_q, and $\sigma = (z, a, b, r)$ is a digital signature from the domain $G_q \times G_q^2 \times G_q^2 \times \mathbb{Z}_q$. A piece of postage is called *valid* if it satisfies the following predicate:

$$\mathit{verifyPoP}(y, A, B, (z, a, b, r)) \equiv \left(G^r \stackrel{?}{=} (ya_1)^c b_1 \ \wedge \ m^r \stackrel{?}{=} (za_2)^c b_2\right) \quad \text{where } c = \mathcal{H}(A, B, z, a, b) \ . \tag{1}$$

[5] We abstract here from essential fault-tolerant mechanisms like typing and (logically) time-stamping messages.

Indicia, in their digital form, are tuples $(A, B, (z, a, b, r), s, rcpt, d/t)$, where $(A, B, (z, a, b, r))$ is a piece of postage, $s \in \mathbb{Z}_q^3$, and the components $rcpt$, d/t denote the recipient and the date and time of the indicium. Additional data about the sender may be included. An indicium is *valid* if it satisfies the following predicate:

$$verifyInd(y, A, B, (z, a, b, r), s, rcpt, d/t) \equiv \left(AB \neq 1 \ \wedge \ g_1^{s_1} g_2^{s_2} G_0^{s_3} \stackrel{?}{=} AB^c\right)$$
$$\text{where } c = \mathcal{H}(A, B, z, a, b, r, rcpt, d/t) \qquad (2)$$

5.1 Opening a Postal Account

Before a sender can open a postal account, she chooses a private digital identity $(u_1, u_2) \in \mathbb{Z}_q^{*2}$ and computes her corresponding public digital identity $I \leftarrow g_1^{u_1} g_2^{u_2} \bmod p$. She identifies herself to the postal services, e.g., by means of a passport, and asks to open an electronic postal account I in her name. She furthermore proves to know a representation of I with respect to the generators g_1, g_2 by means of the following interactive proof of knowledge (Fig. 2). If the postal service accepts her application and finds the protocol *prove* to succeed (acc = TRUE), it binds I to the applicant's name and makes this pair a new entry in its account database.

$$([acc]^P) \leftarrow prove([u_1, u_2, I]^S, [I]^P)$$

	Sender		**Postal Server**
(1)	Choose $w_1, w_2 \in_R \mathbb{Z}_q$;		
	$a \leftarrow g_1^{w_1} g_2^{w_2} \bmod p$	$\xrightarrow{a}$	
(2)		$\xleftarrow{c}$	$c \in_R \mathbb{Z}_q^*$
(3)	$r_1 \leftarrow cu_1 + w_1 \bmod q$		
	$r_2 \leftarrow cu_2 + w_2 \bmod q$	$\xrightarrow{r_1, r_2}$	
(4)			$acc \equiv (g_1^{r_1} g_2^{r_2} \stackrel{?}{=} h^c a \pmod p)$

Fig. 2. Proving knowledge of a representation of I

5.2 Downloading a Piece of Postage

The postage server uses its private key $x \in \mathbb{Z}_q^*$ to provide pieces of postage. If a sender I wants to download a piece of postage from her account, she performs the following interactive protocol with the postage server. The identity I and the postage server's public key y are common inputs, and the postage server uses x as private input. Finally in step (12), the sender can verify the resulting piece of postage by using the verifying predicate (1).

$([A', B', \alpha, \beta, \sigma']^S) \leftarrow download([u_1, u_2, y, I]^S, [x, I]^P)$

	Postal Server		**Sender**
(1)		$\xleftarrow{([acc]^P) \leftarrow prove([u_1, u_2, I]^S, [I]^P)}$	
			proceed iff acc
(2)	Choose $t \in_R \mathbb{Z}_q$		$u \in_R \mathbb{Z}_q^*, v \in_R \mathbb{Z}_q$
(3)	$z \leftarrow (IG_0)^x$		
(4)	$(a, b) \leftarrow (G^t, (IG_0)^t)$	$\xrightarrow{z,a,b}$	
(5)			Choose $\omega \in_R \mathbb{Z}_q, \alpha \in_R \mathbb{Z}_q^3$
			$(I', z') \leftarrow ((IG_0)^\omega, z^\omega)$
(6)			$A' \leftarrow g_1^{\alpha_1} g_2^{\alpha_2} G_0^{\alpha_3}$
			$\beta \leftarrow (u_1\omega - \alpha_1, u_2\omega - \alpha_2, \omega - \alpha_3)$
			$B' \leftarrow g_1^{\beta_1} g_2^{\beta_2} G_0^{\beta_3}$
			{note that now: $A'B' = I' \bmod p$}
(7)			$(a', b') \leftarrow (a^u g^v, b^{\omega u} I'^v)$
(8)			$c' \leftarrow \mathcal{H}(A', B', z', a', b')$
(9)		$\xleftarrow{c}$	$c \leftarrow \frac{c'}{u} \bmod q$
(10)	$r \leftarrow cx + t$	$\xrightarrow{r}$	
(11)			$r' \leftarrow ru + v$
(12)			proceed iff $verifyPoP(y, A', B', (z', a', b', r'))$
(13)			$\sigma' \leftarrow (z', a', b', r')$

Fig. 3. Downloading a Piece of Postage

In step (6), the sender splits the new identity I' into two factors $A', B' \in G_q$ for which she knows respective representations $\alpha_1, \alpha_2, \alpha_3$ and $(\beta_1, \beta_2, \beta_3) = (u_1\omega - \alpha_1, u_2\omega - \alpha_2, \omega - \alpha_3)$ with respect to g_1, g_2, G_0.

5.3 Making Indicia

When the sender wants to send some mail, she chooses a suitable piece of postage $(A, B, \alpha, \beta, \sigma)$ and computes the corresponding indicium[6]. The computation depends on the recipient *rcpt*, the actual date and time *d/t* and possibly other postal relevant data items (see Subsection 6.5). In addition to the piece of postage, the sender also needs to input the representations α, β of A and B, respectively (Figure 4).

5.4 Detecting Double Spenders

When a piece of mail passes the mail delivery system, the respective indicium can be verified according to predicate (2).

[6] Note that in the previous subsection, pieces of postage have been denoted as $(A', B', \alpha, \beta, \sigma')$, where the primes are used for technical reasons. From here on we drop the primes

$(s) \leftarrow \mathit{indicium}(A, B, \alpha, \beta, (z, a, b, r), \mathit{rcpt}, d/t)$

Sender

(1) $c \leftarrow \mathcal{H}(A, B, z, a, b, r, \mathit{rcpt}, d/t)$

(2) $s = (s_1, s_2, s_3)$, where $s_i \leftarrow \alpha_i + c\beta_i$ for $i = 1, 2, 3$

Fig. 4. Computing an Indicium

When a customer uses any downloaded piece of postage $(A, B, (z, a, b, r))$ to print out more than one indicium, then the mail delivery system can detect the fact of "double spending" by simply checking whether the components (A, B) have occured in previous indicia. If so, we denote the challenges of the two indicia as c_1, c_2 (cf. step (1)) in Fig. 4 and the corresponding s-components as $s_1 = (s_{11}, s_{12}, s_{13})$ and $s_2 = (s_{21}, s_{22}, s_{23})$. (Note that each s_i is a triple of numbers.) Then the mail delivery system can figure the double spender's private digital identity (u_1, u_2) as follows (Figure 5):

$(u_1, u_2) \leftarrow \mathit{identifyDS}(c_1, s_1, c_2, s_2)$

Mail Delivery System

(1) $(u_1, u_2) \leftarrow \left(\frac{s_{11}(c_2-1)+s_{21}(1-c_1)}{s_{13}(c_2-1)+s_{23}(1-c_1)} \bmod q, \ \frac{s_{12}(c_2-1)+s_{22}(1-c_1)}{s_{13}(c_2-1)+s_{23}(1-c_1)} \bmod q \right)$

Fig. 5. Identifying a Double Spender

5.5 Remarks on Elliptic Curve Implementation

Koblitz has shown a very efficient method that picks a suitable curve of prime order q and a point P at the same time [K87]. In this case, the elliptic curve is cyclic and any point (except infinity) is a generator. Once, a generator P on the curve is known, it is easy to compute more generators uniformly at random by multiplying P by a randomly chosen integer $a \in \mathbb{Z}_q$.

5.6 Security

The proposed implementation of an offline e-cash scheme follows ideas of Brands' [B93,B94]. Applying his results ensures that the above scheme is effective and meets UNFORGEABILITY, DOUBLE SPENDING DETECTION, UNFRAMEABILITY and INDICIA UNLINKABILITY. It also achieves OFFLINE PRINTING because indicia can

be produced and printed by the sender alone (see algorithm *indicia*). Unlike Brands' proposal using tamper resistant hardware to prevent double spending, we only require that each indicium contains the recipient's name, address or ZIP and the date and time of producing the indicium. This works because these specific data items are known to the mail delivery system anyway and can therefore be used to verify indicia.

6 Design and Performance Options

6.1 Multiple Denominations

The above scheme handles only one type of piece of postage, i.e., one denomination of postage, and at the same time, it handles only one denomination of indicia. In practice, a small number of different denominations are desirable, e.g., 24c, 33c, 60c, 1$, etc., and this set of denominations changes every time postage rates are changed. One way to implement several denominations is to use a number of different key pairs for the postage server. Each key then corresponds to one denomination. An alternative was proposed by Brands. He suggests to represent denominations in binary and to use a separate generator analogous to G_0 for each binary digit of the denomination. For example, if all denominations between 1 and 255 cents may occur and we like to express a denomination of $v = 60_{10}$ cents, then the Postal Server can use eight generators $G_0, \ldots, G_7$ instead of only G_0. In steps (3) and (4) of Figure 3 the Postal Server uses the product $I \prod_{i=0}^{7} G_i^{v_i}$ instead of IG_0 to build components z and b. Note that $(v_7, \ldots, v_0) = (0, 0, 1, 1, 1, 1, 0, 0)$ is the binary representation of v. Analogously, the Sender chooses A', B' in step (6) so that she can represent them according to $G_7, \ldots, G_0$.

The first alternative is preferable if the set of necessary denominations at each time is relatively small. If the set is larger, then an equally large set of public keys of the Postal Server must be managed. In this case, some additional computing effort of the Postal Server (steps (3) and (4)) and of the Sender (step (6)) during download is justified. The number of additional exponentiations is $\log m$ for the postal server and $2 \log m$ for the sender, where m is the maximum denomination.

6.2 Divisibility

With the above proposal, senders can download any number of pieces of postage of any denominations available at the postage server. A straightforward implementation allows to use one at a time to print an indicium. It is more convenient though if senders can produce indicia of denominations not necessarily the same as those of the pieces of postage they have downloaded in the first place. One option is to combine more than one denomination in one indiciumsimilar to conventional stamps, which can be combined on an envelope. This is straightforward and combinations are limited only by the bit capacity of printable indicia. The above back of the envelope calculation indicates that there is probably room for

more than one elliptic curve signature of the proposed system even in a 1 square inch indicium. If that is not enough, 2 by 1 inch indicia might also be acceptable.

Another option is to obtain divisible pieces of postage that a sender can download as a whole and later decide how to split them up. A solution has been proposed by Brands[B93] for the e-cash scheme underlying our proposal in Section5.

6.3 Graphical Requirements of Indicia

Indicia in digital form are tuples $(A, B, (z, a, b, r), s, rcpt, d/t)$ of 5 elements from G_q, namely A, B, z, a, b, 4 elements of $\mathbb{Z}_q$, namely r, s_1, s_2, s_3, and further data items such as $rcpt, d/t$) each 32 bit long. If p is a 1024 bit prime, q at least 160 [O99] bit prime divisor of $p-1$ and G_q is a subgroup of $\mathbb{Z}_p^*$, then the digital representation of an indicium is 5904 bits = 738 bytes long. Add 50 bytes of plaintext for recipient address or ZIP, date and time, etc. [IBIP99o], then an indicium must encode about 788 bytes. If a Data Matrix code of RVSI Acuity CiMatrix (figure 1) is used (including suitable Reed-Salomon error correction ECC200) [DM96], an indicium can be printed in one square inch at a resolution of about 104 rows $\times$ 104 columns per inch, which corresponds to $4 \times 104 = 416$ dpi. (Note that rows and columns need to be at least 3-4 dots wide, in order for black unit elements to appear as squares rather than circles.) Alternatively, if G_q is a suitable elliptic curve of an order the size 160 bit, i.e., where computing discrete logarithms is assumed to be as hard as in the former example, then the digital representation of an indicium is 1504 bits = 188 bytes long. Add again 50 bytes of plaintext, and an indicium must encode 238 bytes. Using the Data Matrix code above, such an indicium can be printed in one square inch at a minimum resolution of about $4 \times 128 = 256$ dpi. This appears perfectly realistic to reproduce with a standard inkjet printer at 300 dpi, let alone with a laser printer. For comparison, to implement IBIP compliant indicia using ECDSA or RSA signatures requires a minimum resolution of about $4 \times 40 = 160$ dpi or $4 \times 52 = 208$ dpi respectively (see Table 1).

Table 1. Minimal Print and Scan Resolutions

		size of indicium	min. resolution
New proposal	over finite field	738 byte	416 dpi
	over elliptic curve	238 byte	256 dpi
IBIP compliant	RSA	178 byte	208 dpi
	ECDSA	90 byte	160 dpi

6.4 Storing and Refunding Pieces of Postage

Senders will usually feel more comfortable if they need not rely on their PC hard-drives alone to store their pieces of postage. Regular backups by tape, diskette

or Palmtop are the most conventional solution. Where smartcard readers are available, smartcards can be used as a handy storage medium. Palmtops that provide sufficient computing power and a printer interface on their own, could also serve as a postage wallet themselves.

Pieces of postage that are not needed any longer can be returned to the postage server by transmitting the indicia data electronically instead of printing them out. Indicia printed in error can be sent in by conventional mail.

6.5 Copy Protection

The above proposal requires a huge distributed database in which the postal services store and maintain all past indicia that are still not expired. Each time an indicium is scanned off of a mailpiece it needs to be looked up in the database. If it is found, then it is either identical to one that has been scanned before, or it originates from the same piece of postage as another indicium that has been scanned before. In the former case, an exact copy is found. In the latter case, the attacking customer can be identified using Figure 5.

From the point of view of an attacker, Xerox copies of indicia are a no-risk game. If they go undetected, the attacker has sent mail without paying, otherwise at worst the mail is not delivered. This problem can be reduced by reading a fingerprint of a mailpiece and including it in the indicium. In our solution, the fingerprint can be used in addition to the recipient address and the time/date stamp to produce the indicium (see operation indicium in Figure 4). Recent work of Joshua Smith (`http://www.escherlabs.com`) uses the ridges at the surface of a paper envelope to produce a unique fingerprint. This concept requires that each customer must have additional equipment to scan the fingerprint of his envelopes, and so must the mail delivery system be equipped. It is important to note that it may be acceptable to apply a unique fingerprint to an envelope or box by means of a label. In this case, an attacker can use a Xerox copy of an indicium only if he removes the fingerprinting label from the original envelope to re-apply it to the copy envelope. This might be acceptable because the indicium works only for a given address ZIP code within a given time limit. The relabeling attack works only if a sender S_1 mails an envelope with fingerprint label L to a recipient R, R physically returns the fingerprint label L together with the indicium back to a sender S_2, which may be different from S_1, and finally S_2 applies label L to a new envelope, copies the indicium onto this new envelope and mails it to R within the time limit of the indicium. If the lifetime of an indicium is strict enough, this burden for an attacker is probably not worth the gain. In this case various hard to copy label technologies could be used such as holograms, watermarks, imprints, etc.

7 IBIP in a Nutshell

The IBIP Program specifies an architecture for open and closed franking systems [IBIP99o,IBIP99c]. Both approaches are very similar and totally different from the proposal above. In a nutshell, the IBIP architecture is as follows.

Each customer basically holds a physical tamper responsive wallet, which stores all postage for its customer. This wallet is called postal security device (*PSD*) and it is intended to remain seated in the customer's PC (open system) or meter machine (closed system). Basically, each PSD houses a postage counter and a private signature key, which is used to produce indicia. During download, a PSD's postage counter is incremented. Upon command of its customer and if sufficient postage value is indicated by the internal counter, the PSD produces an indicium by signing the postage value, the PSD ID, the sender's ZIP code, the postage counter content and a number of other data items with the PSD's individual indicium signing key. This data and signature are then encoded into a 2-D bar code, which is easy and reliable to scan by the mail delivery system when printed. Finally, the internal postage counter is decremented by the amount of postage printed. The main difference between the open and closed system architecture is that for open systems, indicia must contain the recipient address or ZIP, whereas for closed systems they need not. This reveals the main advantages of the proposed software solution:

1. No crypto device: Customers can simply use their normal PCs, Laptops or PalmTops. No additional PSD, is needed. For cost reasons, mass market availability and tamper responsiveness of PSDs are contradictory requirements. Striving for both means to constantly maintain the fragile balance between cost and security, which probably results in frequent design changes and little customer satisfaction and acceptance.
2. No FIPS certification of customer software: Customers are not required to use any private key for a public or secret key cryptosystem, and therefore the customer software needs no FIPS certification.
3. Minimal PKI: The senders and the mail delivery system only need to know the postage servers' public key(s). Both can lookup these keys at the postal services' webpage, and the respective Postal Services' certificates can be preloaded into the standard web browsers. In contrast, the IBIP Specification requires senders to use their individual indicium signing keys, which implies a significant amount of public key management.
4. No hardware assumption: The key assumption underlying the IBIP specification is that PSD's cannot be tampered with. IBIP requires PSDs be FIPS 140-1 (level 3-4) certified. If an attacker should break a PSD anyway, he can forge any amount of postage, and it will be very hard to even detect the fraud, let alone to identify the attacker. In contrast, the key assumption underlying the above software solution is a cryptographic assumption, which has been established in 1991 by Chaum, Pedersen [CP92] and Brands [B94] and has not been defeated since. The fundamental difference between a cryptographic and a hardware assumption is that a cryptographic assumption can be formally stated if not formally proven. A hardware assumption however is principally unprovable and tends to invoke an arms' race between those looking for new hardware implementations they believe meet the assumption and those who penetrate existing hardware implementations.

As a result of these advantages, the cost of development, production and running of the proposed software solution should be significantly below any IBIP compliant implementation. The costly constraints imposed by IBIP [IBIP99o] on a franking system architecture serve primarily the security interests of the postal services, but must be paid for by the customers. Our software based solution is probably more competitive because it serves the security requirements of both postal services and customers better and reduces the price tag for the customer. It is conceivable that this software solution escapes the low-cost-high-security dilemma mentioned in the introduction.

8 Conclusion

We have proposed a software solution for open franking systems that is an extension of a well-known offline e-cash scheme proposed by Brands [B94] and later used by the ESPRIT project Cafe [BBCM94]. The new solution also applies to low-end postage meter machines that include the recipient address or ZIP within the indicia. The new software solution is a) unforgeable, b) effectively discourages double spending of downloaded postage, c) detects double use after the fact, and gives c) unframeability and d) indicia unlinkability to the customer. Unforgeability and double spending detection rely on a cryptographic assumption, which stands since about a decade. The main advantages of this software solution over the emerging IBIP Specification of the USPS are as follows: No tamper responsive hardware is needed at the customer's site, no full-blown public key infrastructure is needed because customers do not use individual public keys, and different levels of customer privacy up to strong indicia unlinkability can be provided.

The proposed software solution also reveals that the IBIP Specifications [IBIP99o,IBIP99c] by the US postal services are essentially an overspecification. They prescribe a system architecture down to a level of detail such that any implementation meeting the IBIP Specification is inherently vulnerable to hardware tampering. Our software solution shows that the high level requirements of IBIP can be met by a non-compliant implementation that does not observe the above vulnerability and in addition allows to give more privacy to the customers. In terms of cost, IBIP compliant solutions are very likely to be more expensive than the software solution proposed.

9 Acknowledgment

It is a pleasure for me to thank Andreas Wagner and Dieter Pauschinger for introducing me to franking systems and for supporting this work in particular. Dahlia Malkhi made interesting comments on an early version of this work.

References

[B93] Stefan Brands: An Efficient Off-line Electronic Cash System Based On The Representation Problem; Centrum voor Wiskunde en Informatica, Com-

puter Science/Departement of Algorithmics and Architecture, Report CS-R9323, March 1993.

[B94] Stefan Brands: Untraceable Off-line Cash in Wallet with Observers; Crypto '93, LNCS 773, Springer-Verlag, Berlin 1994, 302-318.

[B96] Stefan Brands: Privacy-Protected Transfer of Electronic Information; United States Patent 5,521,980, issued 05/28/1996.

[BBCM94] Jean-Paul Boly, et al: The ESPRIT Project CAFE High Security Digital Payment Systems; ESORICS 94 (Third European Symposium on Research in Computer Security), Brighton, LNCS 875, Springer-Verlag, Berlin 1994, 217-230.

[BC97] Stefan Brands, David Chaum: 'Minting' electronic cash; IEEE Spectrum 34/2 (1997) 30-34.

[CHP91] David Chaum, Eugne van Heijst, Birgit Pfitzmann: Cryptographically Strong Undeniable Signatures, Unconditionally Secure for the Signer; Crypto '91, LNCS 576, Springer-Verlag, Berlin 1992, 470-484.

[CP92] David Chaum, Torben Pryds Pedersen: Wallet Databases with Observers. Crypto '92, LNCS 740, Springer Verlag, Berlin 1993, 89-105.

[CP94] Ronald J. F. Cramer, Torben P. Pedersen: Improved Privacy in Wallets with Observers (Extended Abstract); Eurocrypt '93, LNCS 765, Springer-Verlag, Berlin 1994, 329-343.

[DM96] ANSI/AIM BC11: International Symbology Specification - Data Matrix; 11/96, http://www.rvsi.com/cimatrix/DataMatrix.html

[FS87] Amos Fiat, Adi Shamir: How to Prove Yourself: Practical Solutions to Identification and Signature Problems; Crypto '86, LNCS 263, Springer-Verlag, Berlin 1987, 186-194.

[IBIP99c] The United States Postal Services: Information-Based Indicia Program (IBIP)—Performance Criteria For Information-Based Indicia and Security Architecture For Closed IBI Postage Metering Systems (PCIBI-C); Draft January 12, 1999, http://ibip.tteam.com/html/programdoc.html

[IBIP99o] The United States Postal Services: Information-Based Indicia Program (IBIP)—Performance Criteria For Information-Based Indicia and Security Architecture For Open IBI Postage Evidencing Systems (PCIBI-O); Draft June 25, 1999, http://ibip.tteam.com/html/programdoc.html

[IEEE] IEEE P1363: Standard Specifications for Public-Key Cryptography; http://grouper.ieee.org/groups/1363/

[K87] Neal Koblitz: A Course in Number Theory and Cryptography; Graduate Texts in Mathematics GTM 114, Springer-Verlag, Berlin 1987.

[SN92] Sebastiaan von Solms, David Naccache: On Blind Signatures and Perfect Crimes; Computers & Security 11/6 (1992) 581-583.

[MOV97] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone: Handbook of Applied Cryptography; CRC Press, Boca Raton 1997.

[O99] Andrew Odlyzko: Discrete Logarithms: The Past and the Future; Designs, Codes, and Cryptography (1999). http://www.research.att.com/ amo/doc/discrete.logs.future.ps

[P91] José Pastor: CRYPTOPOST, A Cryptographic Application to Mail Processing; Journal of Cryptology 3/2 (1991) 137-146.

[S91] Claus P. Schnorr: Efficient Signature Generation by Smart Cards; Journal of Cryptology 4/3 (1991) 161-174.