How to Make Efficient Fail-stop Signatures

Eugène van Heyst

CWI Centre for Mathematics and Computer Science,
Kruislaan 413, 1098 SJ Amsterdam, The Netherlands.

Torben Pryds Pedersen[‡]

Aarhus University, Computer Science Department,
Ny Munkegade, DK-8000 Århus C, Denmark.

Abstract. Fail-stop signatures (introduced in [WP89]) have the very nice property that the signer is secure against unlimited powerful forgers. However, the known fail-stop signatures require very long keys, and they are quite inefficient, because messages are signed bit-wise. This paper presents a fail-stop signature scheme, in which signing a message block requires two modular multiplications and verification requires less than two modular exponentiations. Furthermore a construction is shown of an undeniable signature scheme, which is unconditionally secure for the signer, and which allows the signer to convert undeniable signatures into fail-stop signatures. This is the first published undeniable signature having this property.

1. Introduction

Digital signatures, as introduced in [DH76], allow a person who knows a secret key to make signatures that everybody can verify. These signatures are only computationally secure in the sense that a forger with unlimited computing power can always make false signatures of other persons. Hence, the security of these systems relies on the fact that a forger has not enough time and no efficient algorithm to carry out the computations needed to forge signatures.

With fail-stop signatures (see [WP89]), unforgeability also relies on a cryptographic assumption, but if nevertheless a signature is forged, then the presumed signer can prove that the signature is a forgery: he can prove that the underlying assumption of the system has been broken. This proof of forgery may fail with a very small probability, but the ability to prove a forgery does not rely on any cryptographic assumption and is independent of the computing power of the forger. Hence, the polynomially bounded signer is protected against all powerful forgers, because after the first forgery, all other participants in the system and the system operator know that the signature scheme has

[‡] Research done while visiting CWI.

R.A. Rueppel (Ed.): Advances in Cryptology - EUROCRYPT '92, LNCS 658, pp. 366-377, 1993.

been broken, and the system will be stopped. That is why this system is called "fail-stop".

Fail-stop signatures are known to exist if claw-free permutation pairs exist (see for instance [BPW90] and [PW91]). These signatures use the idea of the one-time signatures of [La79]. In particular, this means that the signatures and keys are very long, and signing as well as verification require quite a lot of computation. Several tricks are therefore applied to make the signatures more efficient (see for example [PW91]).

However, these tricks are not needed in one of the applications of fail-stop signatures, called the 3-phase protocol, proposed in [PW91]. Here it is suggested that customers in an electronic payment system should use fail-stop signatures when signing a request to the bank (for example for the withdrawal of money). These signatures have the advantage over usual digital signatures that the customers need not worry about the bank (which in general has more computing power than the customer) being able to break the underlying assumptions of the signature scheme. In this 3-phase protocol for making such requests, the customer only has to sign one single bit with a fail-stop signature. Hence, this protocol is quite efficient, but it is interactive, and it relies on the assumption that the set of possible recipients is fixed.

It has been very unsatisfactory that the "general", known fail-stop signatures were much less efficient than digital signatures (see [PW91]).

The main objective of this paper is to present a fail-stop signature scheme whose complexity is comparable with that of RSA-signatures (see [RSA78]). More specifically, for each message the public key is approximately 1000 bits (assuming a modulus of size 500 bits), the secret key is approximately 2000 bits and the signature of a 500-bit message is 1000 bits (a longer message can be hashed to 500 bits before signing). The construction of a signature requires two multiplications and two additions modulo a prime, and the verification requires only a little more thane one exponentiation modulo a prime. Thus messages can be signed very efficiently, whereas the verification of signatures is a little slower than for RSA-signatures.

In the next section, we give an informal description of fail-stop signatures and discuss the properties of such signatures. After a brief description of the notation, Section 4 presents the new scheme and it is proven that it satisfies the requirements of fail-stop signatures. The scheme presented here can only be used to sign a single message, but in Section 5 it is shown how to generalize this scheme to sign a constant number of messages. Section 6 uses the ideas of Section 4 to construct an undeniable signature scheme which is unconditionally secure for the signer. This scheme has the property that the signer can convert these undeniable signatures into fail-stop signatures. Finally, Section 7 discusses some applications of the presented schemes.

2. Fail-stop Signatures

This section gives a brief and rather informal description of the properties of fail-stop

signatures, which are required for the scheme in Section 4 (for a formal definition see [PW90]).

In a fail-stop signature scheme, (exponentially in a security parameter) many secret keys correspond to a given public key, and different secret keys will (with very high probability) give different signatures on the same message. However, the signer knows only one of these secret keys and can therefore construct only one of these signatures on a message. A signature is called *valid* if it passes the public test function.

Furthermore, given the public key and signatures on some messages, a forger must not be able to guess which signature the signer is able to construct on a new message, so that even if the forger (by using his unlimited power) succeeds in making a valid signature, this signature will with very high probability be different from the signer's signature. Given such a forged signature the signer must then be able to prove that it is different from his own signature, thereby proving that it is a forgery. After having discovered such a forgery and proved it, the signer should stop using the scheme.

More specifically, if SK is the secret key of the signer, and PK the public key, the signature S on a message, m, is denoted by $S=sign(SK, m)$. The recipient of such a signature can verify its correctness using the polynomial time computable predicate $test(PK, m, S)$ (here polynomial time means polynomial in a security parameter k). For fail-stop signatures this predicate must satisfy that for every secret key SK^* corresponding to PK

$$test(PK, m, sign(SK^*, m)) \text{ is } true.$$

A scheme is a fail-stop signature scheme if it satisfies the following three requirements:

(i) Let PK and the signature $S = sign(SK, m)$ on m be given. Then there are exponentially many (in k) possible secret keys SK^* corresponding to PK such that $S = sign(SK^*, m)$. Furthermore, if such a secret key SK^* is chosen at random, then the probability that $sign(SK, m^*) = sign(SK^*, m^*)$ is negligible, for every message $m^* \neq m$.
(Informally: it is not possible to compute the signer's signature on a new message, even with unlimited computing power.)

(ii) There is a polynomial-time computable function *proof*, which on input SK, PK, a message m, and a valid, forged signature $S' \neq sign(SK, m)$ on m, outputs a proof that S' is a forgery.
(Informally: the presumed signer is able to supply a proof of the forgery.)

(iii) No signer with polynomial-time computing power is able to construct a valid signature S on a given message m and also construct a proof that S is a forgery.
(Informally: the signer cannot make signatures which he can later prove to be forgeries.)

These requirements are for one-time keys. It is not hard to generalize this definition of fail-stop signatures to comprise schemes in which more than one message can be signed.

The first two requirements imply that fail-stop signatures are unconditionally secure for the signer, whereas the third requirement says that the scheme is secure for the

recipients of the signatures. Unlike the security for the signer, the security for the recipient depends on a complexity theoretic assumption. The reader is referred to [PW90] for a thorough discussion of the properties of fail-stop signatures. (To avoid confusion, note that [PW90] considers different security parameters for the security of the signer and for the security of the recipient, while in our scheme these are equal.)

3. Notation

Throughout this paper p and q denote large primes such that q divides $p-1$, and G_q is the unique subgroup of $\mathbb{Z}_p^*$ of order q. As any element $b \neq 1$ of G_q generates the group, the discrete logarithm of $a \in G_q$ with respect to the base b is defined, and it is denoted by $\log_b(a)$.

4. Signature Scheme

In this section an efficient fail-stop signature based on the discrete logarithm assumption is described. Let g and h be elements of G_q such that no participant knows $\log_g(h)$. These elements can either be chosen by a trusted authority, when the system is initialized, or by some of the participants using a coin-flipping protocol.

Although (p, q, g, h) is part of the public key, it will not be mentioned in the following. To give a better idea of the scheme, we will in this section assume that a person issues only one signature. Let the secret key of person $\mathcal{A}$ be

$$SK = (x_1, x_2, y_1, y_2) \in \mathbb{Z}_q^4,$$

and $\mathcal{A}$ publishes the corresponding public key

$$PK = (p_1, p_2) = (g^{x_1} h^{x_2}, g^{y_1} h^{y_2}). \tag{1}$$

To sign a message $m \in \mathbb{Z}_q$, $\mathcal{A}$ computes the following numbers:

$$S = sign(SK, m) = (\sigma_1(SK, m), \sigma_2(SK, m)), \text{ where} \tag{2}$$
$$\sigma_1(SK, m) \equiv x_1 + m y_1 \pmod q,$$
$$\sigma_2(SK, m) \equiv x_2 + m y_2 \pmod q.$$

The recipient of this signature verifies that

$$p_1 p_2^m = g^{\sigma_1} h^{\sigma_2}. \tag{3}$$

The proof of forgery is $\log_g(h)$.

The following three lemmas show that this signature scheme is a fail-stop signature scheme. First note that for every secret key, SK^*, corresponding to PK, the predicate $test(PK, m, sign(SK^*, m))$ is *true* for all messages, m, i.e., for every tuple $(x_1, x_2, y_1, y_2) \in \mathbb{Z}_q^4$ that satisfies

$$PK = (g^{x_1}h^{x_2}, g^{y_1}h^{y_2}),$$
$$\sigma_1(SK, m) \equiv x_1 + my_1 \pmod q,$$
$$\sigma_2(SK, m) \equiv x_2 + my_2 \pmod q,$$

we have

$$p_1 p_2^m = g^{\sigma_1} h^{\sigma_2}.$$

Lemma 1. *The public key PK, together with a signature sign(SK, m) on m, contain no information about which of q possible secret keys are used for SK.*

Proof. This lemma is a special case of Thm 4.4 of [Ped91]. Another way to prove it is the following. Define $h = g^a, p_1 = g^{e_1}, p_2 = g^{e_2}$. This representation is possible because g is a generator of G_q. Then we can write Equations (1) and (2) as:

$$\begin{cases} e_1 \equiv x_1 + ax_2 \pmod q, \\ e_2 \equiv y_1 + ay_2 \pmod q, \\ \sigma_1 \equiv x_1 + my_1 \pmod q, \\ \sigma_2 \equiv x_2 + my_2 \pmod q. \end{cases}$$

The fact that equation (3) holds, follows immediately from equations (1) and (2) as noted above. The forger has to find a solution (x_1, x_2, y_1, y_2) to the Equations

$$\begin{pmatrix} 1 & a & 0 & 0 \\ 0 & 0 & 1 & a \\ 1 & 0 & m & 0 \\ 0 & 1 & 0 & m \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ y_1 \\ y_2 \end{pmatrix} \equiv \begin{pmatrix} e_1 \\ e_2 \\ \sigma_1 \\ \sigma_2 \end{pmatrix} \pmod q.$$

It is easy to see that this matrix has rank 3 (the rank is defined because q is prime), hence, since there is one solution, there are exactly q solutions to this equation. ❑

Lemma 2. *Let PK, the signature $S = sign(SK, m)$ on m and a valid signature $S' = (\tau_1, \tau_2)$ on $m' \neq m$ (so $p_1 p_2^{m'} \equiv g^{\tau_1} h^{\tau_2}$), be given. Then there exists a unique secret key, SK*, corresponding to PK such that $S = sign(SK^*, m)$ and $S' = sign(SK^*, m')$*

Proof As in the proof of Lemma 1, a solution (x_1, x_2, y_1, y_2) to the matrix equation

$$\begin{pmatrix} 1 & a & 0 & 0 \\ 0 & 0 & 1 & a \\ 1 & 0 & m & 0 \\ 0 & 1 & 0 & m \\ 1 & 0 & m' & 0 \\ 0 & 1 & 0 & m' \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} e_1 \\ e_1 \\ \sigma_1 \\ \sigma_2 \\ \tau_1 \\ \tau_2 \end{pmatrix}$$

has to be found. It is easy to see that this matrix has rank 4 (because $m' \neq m$), so there is exactly one solution. ❑

Lemma 1 says that there are q possible secret keys corresponding to a given public key and one signature. By Lemma 2, each of these will yield a different signature on a message $m' \neq m$. This shows that the first requirement for the security of the signer is

satisfied.

Lemma 3. *If the presumed signer receives a valid, forged signature* $S' = (\tau_1, \tau_2)$ *on* m *(so* $p_1 p_2^m \equiv g^{\tau_1} h^{\tau_2}$*), but* $S' \neq sign(SK, m)$*, then he can compute* $\log_g(h)$.

Proof By writing $S = sign(SK, m) = (\sigma_1, \sigma_2)$, we have that $p_1 p_2^m \equiv g^{\tau_1} h^{\tau_2} \equiv g^{\sigma_1} h^{\sigma_2}$ and thus that $g^{\sigma_1 - \tau_1} \equiv h^{\tau_2 - \sigma_2} \pmod{p}$. If $\sigma_2 = \tau_2$, then we also have that $\sigma_1 = \tau_1$ and thus $S = S'$. This is a contradiction, and therefore the presumed signer can compute $\log_g(h)$ as $(\sigma_1 - \tau_1)(\tau_2 - \sigma_2)^{-1} \bmod q$. ❑

Hence, under the assumption that the signer cannot compute $\log_g(h)$, he is not able to construct a valid signature and also a proof that it is a forgery. Thus we can define

$$proof(SK, S') := (\sigma_1 - \tau_1)(\tau_2 - \sigma_2)^{-1} \bmod q.$$

On the other hand the signer cannot compute a proof of forgery without being given a forged signature unless he is able to compute discrete logarithms. This shows that the scheme is computationally secure for the recipients if it is infeasible to compute $\log_g(h)$.

Note that this secret key is a one-time key: if two different messages are signed using the same secret key, then it is easy to compute the secret key from these signatures.

Remark. In the above scheme, the public key consists of two numbers modulo q. It is possible to reduce the size of this public key as follows. Let H be a collision-free hash function that maps the elements of G_q into numbers of a smaller size. Then the public key will be

$$PK^* = (H(p_1), p_2),$$

where p_1 and p_2 are defined as before. A signature (σ_1, σ_2) on the message m is constructed as before, and it is verified as

$$H(g^{\sigma_1} h^{\sigma_2} p_2^{-m}) = H(p_1).$$

By using this public key, the Lemmas 1, 2 and 3 have to be modified. For instance, Lemma 3 has to be modified as follows:

Lemma 3.a. *If the presumed signer receives a valid, forged signature* $S' = (\tau_1, \tau_2)$ *on* m *(so* $H(g^{\tau_1} h^{\tau_2} p_2^{-m}) = H(p_1)$*)), but* $S' \neq sign(SK, m)$*, then he can compute* $\log_g(h)$ *or he has found a collision for* H.

Long messages can first be hashed into smaller messages before signing. In this case a proof of forgery is either $\log_g(h)$ or a collision of the hash function H. Hence,

Lemma 3 has to be modified in a similar way as was done in Lemma 3.a. Even if no hash functions are used in the public key, it is more efficient to verify the signature by computing

$$g^{\sigma_1} h^{\sigma_2} p_2^{-m},$$

and comparing it with p_1. This requires less than $2|q|$ multiplications (where $|q|$ is the number of bits of q), if the products gh, gp_2, hp_2 and ghp_2 are precomputed.

5. More than one Signature per Public Key

As noted in the previous subsection, a person can use his public key (and secret key) only once. We now present three different ways to overcome this problem: the public key can be used to sign k messages, and the signer still has unconditional security. In the first two methods, the secret key consists of $2k$ numbers, while each signature consists of 2 numbers. These two schemes differ in the computations needed. In the third method, the secret key consists of at most k elements, while each signature consists of $\lceil \log k \rceil + 3$ numbers.

Method 1. Person $\mathcal{A}$ chooses as a secret key

$$SK = (x_1, y_1, x_2, y_2, \ldots, x_{k+1}, y_{k+1}),$$

and he publishes the corresponding public key

$$PK = (p_1, \ldots, p_{k+1}) = (g^{x_1} h^{y_1}, \ldots, g^{x_{k+1}} h^{y_{k+1}}).$$

To sign a message $m \in \mathbb{Z}_q$, $\mathcal{A}$ computes the following numbers:

$$sign(SK,m) = (\sigma_1, \sigma_2), \text{ where}$$
$$\sigma_1 \equiv x_1 + mx_2 + \ldots + m^k x_{k+1} \pmod{q},$$
$$\sigma_2 \equiv y_1 + my_2 + \ldots + m^k y_{k+1} \pmod{q}.$$

The recipient of this digital signature verifies that

$$p_1 p_2^m \ldots p_{k+1}^{m^k} \equiv g^{\sigma_1} h^{\sigma_2} \pmod{p}.$$

After issuing signatures on k different messages, the signer still has unconditional security. (This follows from Theorem 4.4 of [Ped91].) Given k signatures, there are q possible secret keys, and $k+1$ signatures determine the secret key uniquely. As before $\log_g(h)$ constitutes a proof of forgery, and it follows from similar arguments that the two last requirements to fail-stop signatures are satisfied.

Method 2. ([Pf91]) Person $\mathcal{A}$ chooses the same secret key and public key as in Method 1. In Method 2, the signature on the message depends on the number of messages that $\mathcal{A}$ has signed previously. If $\mathcal{A}$ has signed $i-1$ messages ($1 \le i \le k$), the signature on a message, m, will be

$$sign(SK, m, i) = (i, \sigma_1, \sigma_2), \text{ where}$$

$$\sigma_1 \equiv x_i + mx_{i+1},$$
$$\sigma_2 \equiv y_i + my_{i+1}.$$

The recipient of this signature verifies that

$$p_i p_{i+1}^m = g^{\sigma_1} h^{\sigma_2}.$$

Hence, at the cost of including a counter in the signatures, the computations of the signer as well as the recipient are easier here than in Method 1.

Again, the security of the recipient follows from the fact that $\log_g(h)$ is a proof of forgery. In order to prove that the signer has unconditional security we first note that any $k+1$ signatures determine the secret key. Hence it is sufficient to show that after issuing k different signatures there are many (q) possible secret keys. This will be done by showing that the rank of the following $(3k+1)\times(2k+2)$ matrix, A_k, is $2k+1$ (remember that $a = \log_g(h)$):

$$A_k = \begin{bmatrix} 1 & 0 & m_1 & & & & & & & \\ & 1 & 0 & m_1 & & & & & & \\ & & 1 & 0 & m_2 & & & & & \\ & & & 1 & 0 & m_2 & & & & \\ & & & & \ddots & & & & & \\ & & & & & 1 & 0 & m_k & & \\ & & & & & & 1 & 0 & m_k & \\ 1 & a & & & & & & & & \\ & & 1 & a & & & & & & \\ & & & & \ddots & & & & & \\ & & & & & & & 1 & a & \end{bmatrix}.$$

This will be done by proving that matrix $\tilde{A}_k$ can be obtained from A_k by elementary row operations, where

$$\tilde{A}_k = \begin{bmatrix} 1 & 0 & m_1 & & & & & & \\ & 1 & 0 & m_1 & & & & & \\ & & 1 & 0 & m_2 & & & & \\ & & & 1 & 0 & m_2 & & & \\ & & & & \ddots & & & & \\ & & & & & 1 & 0 & m_k & \\ & & & & & & 1 & 0 & m_k \\ & & & & & & & 1 & a \end{bmatrix}$$

The matrix $\tilde{A}_k$ clearly has rank $2k+1$. We change A_k to $\tilde{A}_k$ by using Lemma 1 for $i=1,\ldots,k-1$ as follows. Consider the following four rows of the matrix A_k.

$$\begin{array}{cccccccccc} 0 & \ldots & 0 & 1 & 0 & m_i & 0 & 0 & \ldots & 0 \\ 0 & \ldots & 0 & 0 & 1 & 0 & m_i & 0 & \ldots & 0 \\ 0 & \ldots & 0 & 1 & a & 0 & 0 & 0 & \ldots & 0 \\ 0 & \ldots & 0 & 0 & 0 & 1 & a & 0 & \ldots & 0 \end{array}$$

By Lemma 1, this sub-matrix has rank 3 and the third row can be removed. By using this method for $i=1,\ldots,k-1$, we delete all rows (0 ... 0 1 a 0 ... 0), except the last one. The resulting matrix is $\tilde{A}_k$.

Hence, PK and k signatures $sign(SK, m_1, 1),\ldots,sign(SK, m_k, k)$ contain no information about which of q possible secret keys are used for SK, and each of these possible secret keys will yield a different signature on a new message m^*.

Method 3. Use tree-authentication, which is described in [Merk80] and [PW91], for example.

6. Convertible Undeniable Signatures Unconditionally Secure for the Signer

Convertible undeniable signatures were introduced in [BCDP90]. Briefly, these signatures allow the signer of undeniable signatures to change his signatures to ordinary digital signatures. In [CHP91] an undeniable signature scheme was presented in which the signer is unconditionally secure. This section combines the ideas of [BCDP90], [CHP91] and Section 4 by constructing an undeniable signature scheme which is unconditionally secure for the signer and has the property that the signer can convert the undeniable signatures to fail-stop signatures.

Let p, q, g and h be as in Section 2. The secret key of a person $\mathcal{A}$ is

$$SK = (x_1, x_2, y_1, y_2) \in \mathbb{Z}_q^4,$$

and the corresponding public key is

$$PK = (p_1, p_2) = (g^{x_1}h^{x_2}, g^{y_1}h^{y_2}).$$

The undeniable signature of $\mathcal{A}$ on the message $m \in \mathbb{Z}_q$ is

$$\sigma_1(SK, m) \equiv x_1 + my_1 \pmod{q},$$

This signature is undeniable, because given PK, the signature is just a random number in $\mathbb{Z}_q$. The signature scheme is unconditionally secure for the signer, because given σ_1, it is impossible for a forger with unlimited computing power to construct $\sigma_1' = sign(SK, m')$ for a new message $m' \neq m$. This follows from the same arguments as in Lemma 1 and 2. We mention that $\mathcal{A}$ can convert this signature to a fail-stop signature by publishing

$$\sigma_2 \equiv x_2 + my_2 \pmod{q},$$

which changes the undeniable signature into the fail-stop signature of Section 4.

Verification

To verify the signature σ_1 on a message m, $\mathcal{A}$ and the recipient compute $u \equiv p_1 p_2^m g^{-\sigma_1} \pmod{p}$, and $\mathcal{A}$ convinces the recipient with a zero-knowledge protocol that he knows a number σ_2 such that $u \equiv h^{\sigma_2} \pmod{p}$. Perfect zero-knowledge protocols for this problem are well known (e.g. [CEvdG87]), hence

Theorem 4. *There is a perfect zero-knowledge protocol for convincing someone of having* σ_2 *satisfying* $u \equiv h^{\sigma_2} \pmod{p}$.

Disavowal of these signatures is slightly more complicated. A given number $\sigma \in \mathbb{Z}_q$ is not $\mathcal{A}$'s signature on m if

$$p_1 p_2^m \not\equiv g^{\sigma} h^{\sigma_2} \pmod{p},$$

where $\sigma_2 \equiv x_2 + m y_2 \pmod{q}$. $\mathcal{A}$ can therefore convince someone that σ is not his signature by convincing him that he knows numbers s and t such that

$$p_1 p_2^m \equiv g^s h^t \pmod{p}, \quad \text{and} \quad \sigma \neq s.$$

(because if σ was his signature this would mean that $\mathcal{A}$ knew $\log_g(h)$). A perfect zero-knowledge proof for this was presented in [CHP91].

7. Applications

As mentioned in the introduction, fail-stop signatures have been suggested to be used in electronic payment systems, such that the customer does not need to rely on the assumption that the bank does not have sufficient computing power to forge his signatures. Using the previously known schemes, this application required a 3-phase protocol between the bank and the customer in which also usual digital signatures are used (see [PW91]). If the signature scheme presented here is used, this protocol can be avoided and the customer just needs to send a single message to the bank in order to sign the request.

8. Conclusion

This paper has described a fail-stop signature scheme which is very efficient from a computational point of view, and in which signatures are only twice as long as the messages (long messages can first be hashed into smaller messages before signing). This scheme makes it possible to avoid the 3-phase protocol of [PW91] when applied to payment systems.

Furthermore, it has been shown how to construct convertible, undeniable signatures, which are unconditionally secure for the signer.

The main disadvantage of the presented schemes is that they can only be used to construct a fixed number of signatures. This property cannot be avoided, because in a forthcoming paper (with Birgit Pfitzmann) it is shown that a signature scheme which is unconditionally secure for the signer, requires secret keys whose lengths are linear in the number of messages to be signed.

Acknowledgements

We wish to thank Birgit Pfitzmann for many discussions about fail-stop signatures and her comments to this paper.

References

[BCDP90] Joan Boyar, David Chaum, Ivan Damgård and Torben Pedersen, Convertible Undeniable Signatures, *Advances in Cryptology-CRYPTO'90*, LNCS 537, Springer Verlag, pp. 189-205.

[CEvdG87] David Chaum, Jan-Hendrik Evertse and Jeroen van de Graaf, An improved protocol for demonstrating possession of discrete logarithms and some generalizations, *Advances in Cryptology -EUROCRYPT '87*, LNCS 304, Springer-Verlag, pp. 127-141.

[BPW90] Gerrit Bleumer, Birgit Pfitzmann and Michael Waidner, A remark on a signature scheme where forgery can be proved, *Advances in Cryptology-EUROCRYPT '90*, LNCS 473, Springer Verlag, pp. 441-445.

[CHP91] David Chaum, Eugène van Heyst and Birgit Pfitzmann, Cryptographically strong undeniable signatures, unconditionally secure for the signer, *Advances in Cryptology -CRYPTO '91*.

[DH76] Whitfield Diffie and Martin Hellman, New directions in cryptography, *IEEE IT* **22** (1976), pp. 644-654.

[FFS88] Uriel Feige, Amos Fiat and Adi Shamir, Zero-Knowledge Proofs of Identity, Journal of Cryptology **1** (1988), pp. 77-94.

[GMR88] Shafi Goldwasser, Silvio Micali and Ronald Rivest, A digital signature scheme secure against adaptive chosen-message attacks, *SIAM J. Comp.* **17** (1988), pp. 281-308.

[La79] L. Lamport, Constructing Digital Signatures from a One-Way Function, SRI Intl. CSL-98 (October 1979).

[Merk80] Ralph C. Merkle, Protocols for public key cryptosystems; *Proceedings of the 1980 symposium on security and privacy*, April 1980, Oakland, California, pp. 122-134.

[Ped91] Torben Pedersen, Non-interactive and information-theoretic secure variable secret sharing, *Advances in Cryptology -CRYPTO '91*.

[Pf91] Birgit Pfitzmann, personal communication.

[PW90] Birgit Pfitzmann and Michael Waidner, Formal aspects of fail-stop signatures, Interner Bericht 22/90, Fakultät für Informatik, Universität Karlsruhe, December 1990.

[PW91] Birgit Pfitzmann and Michael Waidner, Fail-stop signatures and their applications, *SECURICOM '91*, Paris, 1991, pp. 145-160.

[RSA78] Ronald Rivest, Adi Shamir, and Leonard Adleman, A Method for Obtaining Digital Signatures and Public Key Cryptosystems, *Comm. of the ACM* **21** (1978), pp. 120-126.

[WP89] Michael Waidner and Birgit Pfitzmann, The Dining Cryptographers in the Disco: Unconditional Sender and Recipient Untraceability with Computationally Secure Serviceability, *Advances in Cryptology-EUROCRYPT '89*, LNCS 434, Springer Verlag, p. 690.