J. KSIAM **Vol.2,** No.1, 15-26, 1998

A Provably Secure and Practical Signature Scheme for Smart Cards

Yong Kuk You and Sang Geun Hahn

Korea Advanced Institute of Science and Technology,
Department of Mathematics, Taejon 305-701, Korea

Abstract

By "secure", we mean that some well-defined computational assumption can be shown to be sufficient for the scheme not to be existentially forgeable, even under an adaptive chosen message attack. Most, if not all, signature schemes used in practice are based on a computational assumption that is certainly *necessary* for this kind of security, not known to be *sufficient.* Since the work of Goldwasser, Micali and Rivest[**?**], many researches have been done for the secure signature schemes. We modify the Cramer-Damgȧrd scheme to implement a practical and secure signature scheme for smart cards.

1 Introduction

For most digital signature schemes used in practice, it has only been shown that certain plausible cryptographic assumptions, such as the collision-intractibility of factoring integers, computing discrete logarithms or the collision-intractibility of certain hash functions are necessary for the securities of the schemes, while their sufficiency is an open question.

A clear advantage of such schemes over many signature schemes with security proven relative to such common cryptographic assumptions, is their efficiency. We want to design a digital signature scheme that offers *both* high security and practical value. First, relative to some plausible cryptographic assumption, a proof must be given that the scheme is not existentially forgeable under adaptively chosen message attacks [**?**]. A signature scheme is *existentially unforgeable* if, given any polynomial (in the security parameter) number of pairs

$$(m_1, S(m_1)), (m_2, S(m_2)), \cdots, (m_k, S(m_k)),$$

where $S(m)$ denotes the signature on the message m, it is computationally infeasible to generate a pair $(m_{k+1}, S(m_{k+1}))$ for any message $m_{k+1} \notin \{m_1, \cdots, m_k\}$, even if it

is random or nonsensical. And an *adaptive chosen message attack* means that the enemy can use the signer $\mathcal{A}$ as an "oracle"; not only may he request from $\mathcal{A}$ signatures of messages which depend on $\mathcal{A}$'s public key but he may also request signatures of messages which depend additionally on previously obtained signatures.

Secondly, we require that the amount of computation and the size of the signatures are small, and finally, the amount of storage needed is resonally limited. Benefitting from the special properties of *claw-free trapdoor permutations*, the secure scheme presented in [**?**] achieves signatures of size $O(k \cdot \log i)$, where k stands for a security parameter and i indicates the number of signatures made. Recently, progress has been made in this area. Starting with [**?**], it can be concluded that proven security, moderate amount of computation and provision of any reasonable number of small-sized signatures, can be satisfised [**?**]. Schemes in [**?**], [**?**] are based on RSA-assumptions, but our scheme in this paper is base on the Schnorr scheme.

To present a digital signature that offers both proven security and practical value, we modify the Cramer-Damgård's schnorr-based scheme in [**?**].

2 Preliminaries

2.1 Elliptic Curves over F_{2^m}

In 1985 a variant of discrete log cryptography based on the discrete log problem in the group of points of an elliptic curve defined over a finite field was proposed. These cryptosystems have two potential advantages: (1) the great diversity of elliptic curves available to provide the groups and (2) the absence of subexponential time algorithms that could find discrete logs.

These systems potentially provide equivalent security as the existing public key systems, with shorter key lengths. This can be a crucial factor in some applications, for example the design of smart card systems [**?**]. We will consider only non-supersingular elliptic curves. A non-supersingular curve E over F_{2^m} is the set of all solution to an equation of the form $y^2 + xy = x^3 + ax^2 + b$ with $a, b \in F_{2^m}$, $b \neq 0$ together with the identity O. (It is most convenient to represent O by (0,0).) The following algorithm inputs the points $P_0 = (x_0, y_0)$ and $P_1 = (x_1, y_1)$ on E and returns their sum $P_2 = (x_2, y_2)$.

Algorithm 1. (*Group Operation on E*)

If $P_0 = O$ then output $P_2 \longleftarrow P_1$ and stop
If $P_1 = O$ then output $P_2 \longleftarrow P_0$ and stop
If $x_0 = x_1$
then
if $y_0 + y_1 = x_1$ then output O and stop
else
$\lambda \longleftarrow x_1 + \frac{y_1}{x_1}$
$x_2 \longleftarrow \lambda^2 + \lambda + a$
$y_2 \longleftarrow x_1^2 + (\lambda + 1)x_2$

else

$\lambda \longleftarrow \frac{y_0+y_1}{x_0+x_1}$

$x_2 \longleftarrow \lambda + \lambda + x_0 + x_1 + a$

$y_2 \longleftarrow (x_1 + x_2)\lambda + x_2 + y_1$

Output $P_2 \longleftarrow (x_2, y_2)$

$[-1]P = (x, x + y)$ if $P = (x, y)$. Except for the special cases involving O, the above operations each require 1 multiplicative inversion and 2 or 3 multiplications.

If we let α, β be the roots of the equation $x^2 - tx + q = 0$, where $t = q+1-\#E(F_q)$, then

$$\#E(F_{q^n}) = q^n + 1 - \alpha^n - \beta^n = q^n + 1 - t_n.$$

And $t_n = \alpha^n + \beta^n$ is the sequence satisfying $t_0 = 2, t_1 = t$ and $t_{n+1} = t_n - q^n t_{n-1}$ $(n \geq 1)$.

2.2 Operations on Elliptic Curve $E(F_{2^m})$

Elliptic public-key protocols are based on the operation of $[n]P$, which is called *scalar multiplication* by n. Koblitz introduced a family of curves which admit specially fast elliptic scalar multiplication. This algorithm was later modified by Meier and Staffelbach. We will use the Solinas method[**?**] in this paper.

Squaring

We will assume that the field F_{2^m} is represented in terms of a normal basis over F_2 of the form

$$\{\theta, \theta^2, \theta^{2^2}, \cdots, \theta^{2^{m-1}}\}.$$

Then squarng a field element can be accomplished by a one-bit cyclic shift of the bit string representing the element.

Elliptic Scalar Multiplication

The basic technique is the *addition-subtraction method* . This begins with the *nonadjacent form* (NAF) of coefficient n : a signed binary expansion with the property that no consecutive coefficients are nonzero. For example,

$$NAF(29) =< 1, 0, 0, -1, 0, 1 >$$

since $29 = 32 - 4 + 1$. The average cost of the operation of this way is $\frac{3}{4}m$ elliptic operations.

Anomalous Binary Curves (*ABC*'s)

Two extremely convenient familes of curves are the curves E_0 and E_1 defined over F_2 by

$$E_a : y^2 + xy = x^3 + ax^2 + 1.$$

To avoid attacks such as Pohlig-Hellman Algorithm and Shank's baby step giant step attack, $\#E_a(F_{2^m})$ should be a prime or the product of a prime and small integer.

From the Frobenius map over F_2 :

$$\tau(x, y) = (x^2, y^2),$$

we have the following equation :

$$(\tau^2 + 2)P = (-1)^{1-a}\tau P, \quad \text{for all } p \in E.$$

Since the Frobenius map is just left bit-rotate for the normal basis representation, the multiplication by τ, is essentially free. Thus it is worthwhile, when computing $[n]P$, to regard n as an element of $Z[\tau]$. The following algorithm is for computing the $\tau - addic\ NAF$.

Algorithm 2.($\tau - addic\ NAF$)

Input x_0, y_0
Set $x \longleftarrow x_0, y \longleftarrow y_0$
Set $S \longleftarrow <>$
While $x \neq 0$ or $y \neq 0$,
    If x odd,
        then set $u \longleftarrow 2 - (x - 2y \pmod 4)$
        else set $u \longleftarrow 0$
    Set $x \longleftarrow x - u$
    Prepend u to S
    Set $(x, y) \longleftarrow (y + (-1)^a x/2, -x/2)$
EndWhile
Output S

There is a drawback to this representaton, the $\tau - adic\ NAF$ of an integer n is about twice as long as its ordinary NAF. The solution is the fact that multiplication by τ is done by a one-bit circular shift. So $\alpha, \beta \in Z[\tau]$ with $\alpha \equiv \beta \bmod (\tau^m - 1)$, then $[\alpha]P = [\beta]P$, for all P. Thus the $\tau - adic\ NAF$ of the remainder will have length m, half as long as the $\tau - adic\ NAF$ of n. The ring $Z[\tau]$ is Euclidean with norm function

$$N(x + y\tau) = x^2 + (-1)^{1-a}xy + 2y^2.$$

The following algorithm inputs the dividend $u + v\tau$ and divisor $r + s\tau$ and outputs a quotient $w + z\tau$ and remainder $x + y\tau$ with smaller norm then the divisor.

Algorithm 3.(Division in the Ring $Z[\tau]$)

Input u, v, r, s
Set $k \longleftarrow ru + su + 2sv$,
    $l \longleftarrow rv - su$
Set $h \longleftarrow r^2 + (-1)^{1-a}rs + 2s^2$

Set $w \longleftarrow \lfloor k/h \rfloor$,
$z \longleftarrow \lfloor l/h \rfloor$
Set $x \longleftarrow u - rw - rz - sz$,
$y \longleftarrow v - sw - rz - sz$
Output w, z, x, y

Let $U_0 = 0, U_1 = 1$, and

$$U_k = (-1)^{1-a} U_{k-1} - 2U_{k-2}$$

for $k \geq 2$. Then

$$\tau^m = U_m \tau - 2U_{m-1}.$$

In this way, one can compute $[n]P$ with $\frac{m}{3}$ elliptic operations. Our scheme will not use the $\tau - adic\ NAF$ in the signature generation step, since only required operations are doublings and additions. But the $\tau - adic\ NAF$, will be useful in the verification step.

2.3 Schonorr's Preprocessing Algorithm

The purpose of this preprocessing algorithm is to reduce the computational effort to compute g^r on a week power smart card. Schnorr's original proposal was presented at Crypto'89 [**?**] and cryptanalyzed by de Rooij[**?**]. The following revised algorithm was presented in [**?**], which was also cryptanalyzed by de Rooij[**?**]. We will apply this algorithm to the Cramer-Damgȧrd scheme.

In the followings, p is a large prime, q is a prime that divides $p-1$, and $\alpha \in Z_q$ is a primitive qth root of unity. Suppose that the smart card keeps a collection of k pairs (r_i, x_i), $(0 \leq i < k)$, $r_i \in_R Z_q$, such that $x_i = g^{r_i} \mod p$, and set $v = k$.

1. Pick a random permutation $(a_v(0), \cdots, a_v(k-1))$ of $(0, \cdots, k-1)$, and set $a_v(k) = 0$, $a_v(k+1) = k-1$.

2. Compute the following values:

$$\begin{aligned} r_v^* &= r_{v-k} + 2r_{v-1} \mod q, \\ r_v &= \sum_{i=0}^{k+1} 2^i r_{a_v(i)+v-k} \mod q, \\ x_v^* &= x_{v-k} x_{v-1}^2 \mod p, \\ x_v &= \Pi_{i=0}^{k+1} (x_{a_v(i)+v-k})^{2^i} \mod p. \end{aligned}$$

3. Keep (r_v^*, x_v^*) ready for the next signature. Replace (r_{v-k}, x_{v-k}) with (r_v, x_v).

4. Set $v = v + 1$ and goto step 1. for the next signature.

This requires $2k+2$ multiplications mod p, a storage of k pairs (r_i, x_i) and an update of a pair each time a new pair is an output. De Rooij developed an attack to find the secret key of the signer in $(k!)^2$ steps using $\sqrt{\frac{1}{2}\pi(k-1)k!}$ consecutive signatures. So to

achieve level of 2^{72}, we have to choose $k = 14$.

De Rooij's Attack

The following two equations can be obtained from the signature (y_j, e_j) for $j \geq k$.

$$r_{j-1} = \frac{1}{2}(y_j - se_j - r_{j-k}) \mod q, \quad (1)$$

$$r_j = \sum_{i=0}^{k+1} 2^i r_{a_{v(i)}+j-k} \mod q. \quad (2)$$

By repeated substitution of (2.1) into (2.2), we can obtain, from each signature one equation with $k-1$ unknowns, $r_0, \cdots, r_{k-2}$, and s.

The basic idea of de Rooij's attack is to find two equations in which the unknowns, $r_0, \cdots, r_{k-2}$, have the same coefficients. This then directly gives the secret key s. (See [**?**].)

3 Elliptic Curve Signature Scheme

3.1 Initialization

$\mathcal{M}$ denotes the message space, and two one-way hash functions $\mathcal{H} : \mathcal{M} \longrightarrow Z$, $\mathcal{G} : E(F_{2^m})^3 \longrightarrow Z$ are made public. Signer picks a point $P \in E(F_{2^m})$, with $N = Ord(P)$, and makes them public.

Signer gengerates two independent instances

$$(X, w) \equiv ((X_i, w_1), \cdots, (X_d, w_d)) \quad \text{and}$$
$$(\overline{X}, \overline{w}) \equiv ((\overline{X_1}, \overline{w_1}), \cdots, (\overline{X_d}, \overline{w_d})),$$

with $X_i = [w_i]P$ and $\overline{X_i} = [\overline{w_i}]P$ for $i = 1, \cdots, d$. The roots of the authentication trees, $A_1^1 = [z_1^1]P, A_1^{2s+2} = [z_1^{2s+2}]P, A_1^{2(2s+1)+1} = [z_1^{2(2s+1)+1}]P, \cdots, A_1^{c(2s+1)+1}$ $= [z_1^{c(2s+1)+1}]P$, where $z_1^1, z_1^{2s+2}, z_1^{2(2s+1)+1}, \cdots, z_1^{c(2s+1)+1} \in Z_N$ are precomputed by the following preprocessing with the renewal parameter s and an integer c. And all $z_1^1, z_1^{2s+2} \cdots, z_1^{c(2s+1)+1}$ are different mod N.

$(X, \overline{X}, A_1^1, A_1^{2s+2}, \cdots, A_1^{c(2s+1)+1})$ are placed in the public directory, and $w_i, \overline{w_i}$ for $i = 1, \cdots, d$ are secret. By storing A_1^1 and the differences between consecutive values, this requires hardly any storage. And the smart card only stores $(X, w), (\overline{X}, \overline{w})$ and $(A_b^{i^*}, z_b^{i^*}), b \in \{0, 1\}, 0 \leq i < k$ of the following section.

3.2 Preprocessing over $E(F_{2^m})$

For $(A_b^{i^*}, z_b^{i^*})$, $0 \leq i < k$ such that $A_b^{i^*} = [z_b^{i^*}]P, b \in \{0, 1\}$, set $v = k$.

1. Pick a random permutatoin $(a_v(0), \cdots, a_v(k-1))$ of $\{0, \cdots, k-1\}$ and set $a_v(k) = 0, a_v(k+1) = k-1$.

2. Compute the following values:

$$\begin{aligned} z_b^v &= z_b^{v-k^*} + 2z_b^{v-1^*} \quad \text{mod } N, \\ z_b^{v^*} &= \sum_{i=0}^{k+1} 2^i z_b^{a_v(i)+v-k^*} \quad \text{mod } N, \\ A_b^v &= A_b^{v-k^*} + 2A_b^{v-1^*} \quad \text{over } E(F_{2^m}), \\ A_b^{v^*} &= \sum_{i=0}^{k+1} 2^i A_b^{a_v(i)+v-k^*} \quad \text{over } E(F_{2^m}). \end{aligned}$$

3. Keep (A_b^v, z_b^v) ready for the next signature. Replace $(A_b^{v-k^*}, z_b^{v-k^*})$ with $(A_b^{v^*}, z_b^{v^*})$. (i.e. Keep $(A_b^{i^*}, z_b^{i^*})$, $v-k+1 \le i \le v$.)

4. $v = v+1$ and goto step 1. for the next signature.

3.3 Signature Generation

For a ith message $M^i \in \mathcal{M}$, $1 \le i \le (c+1)s$, with $i = ps+q, 0 \le p \le c, 1 \le q \le s$,

$$M_1^i \parallel \cdots \parallel M_d^i \equiv \mathcal{H}(M^i) \quad \text{mod } dN.$$

First, $r_0^i = z_0^i + M_1^i w_1 + \cdots + M_d^i w_d$. Before establishing an authentication for A_0^i, signer computes $A_1^{p(2s+1)+2q}, A_1^{p(2s+1)+2q+1}$. Next, A_0^i is authenticated, together with $A_1^{p(2s+1)+2q}, A_1^{p(2s+1)+2q+1}$, by computing $r_1^i = z_1^{p(2s+1)+q} + \mu_i\overline{w_1} + \cdots + \mu_d\overline{w_d}$, where

$$\mu_1 \parallel \cdots \parallel \mu_d \equiv \mathcal{G}(A_1^{p(2s+1)+2q} \parallel A_1^{p(2s+1)+2q+1} \parallel A_0^i) \quad \text{mod } dN.$$

Let $\text{Auth}(A_0^i)$ be an authentication path for A_0^i i.e., $\text{Auth}(A_0^i)$ consists of all tuples of the form $(A_1^{p(2s+1)+j}, A_1^{p(2s+1)+2j}, A_1^{p(2s+1)+2j+1}, A_0^{ps+j}, r_1^{ps+j})$, with $1 \le j \le q$, such that $A_1^{p(2s+1)+j}$ is an ancestor of $A_1^{p(2s+1)+q}$. The signature $\sigma(M^i)$ on M^i consists of ($\text{Auth}(A_0^i)$,r_0^i).

We suggested that the authentication tree be renewed after each s time generations of signatures. That is, after each s-th signature, all authentication paths generated before will be removed, and a new authentication path will continue. This is merely for storage. The renewal parameter s and a parameter c will be determined in the consideration of storage of smart card, the size of data transmission and the required number of signatures to be signed respectively. The smart card will generate $(c+1)s$ number of signatures.

3.4 Signature Verification

The verifier puts $\sigma(M^i) \equiv (\text{Auth}(A_0^{j_r}), r_0^{j_r})$, where r is the number of tuples in $\text{Auth}(A_0^i)$ and $(A_1^{j_l}, A_1^{2j_l}, A_1^{2j_l+1}, A_0^{j_l}, r_1^{j_l})$ is the l-th tuple in $\text{Auth}(A_0^{j_r})$.

Verifier checks whether

1. $A_1^{j_1} \in \{A_1^1, A_1^{2s+2}, \cdots, A_1^{c(2s+1)+1}\}$,
2. $A_1^{j_l} \in \{A_1^{2j_{\lfloor \frac{l}{2} \rfloor}}, A_1^{2j_{\lfloor \frac{l}{2} \rfloor}+1}\}$ for $j = 2, \cdots, r$,
3. $[r_0^{j_r}]P = A_0^{j_r} + [m_1]X_1 + \cdots + [m_d]X_d$,
4. $[r_1^{j_r}]P = A_1^{j_r} + [\mu_1]\overline{X_1} + \cdots + [\mu_d]\overline{X_d}$,

with $M_1^i \parallel \cdots \parallel M_d^i \equiv \mathcal{H}(M^i) \bmod dN$, $\mu_1 \parallel \cdots \parallel \mu_d \equiv \mathcal{G}(A_1^{2j_r} \parallel A_1^{2j_r+1} \parallel A_0^{j_r}) \bmod dN$.

3.5 Security

First the original scheme of Cramer-Damgård[**?**] is secure in the sense that under an adaptive chosen message attack, it is not existentially forgeable. But its signature size is $O(k \log i)$ bits, where k is the number of bits needed to represent an element, and i indicates the number of signatures made, so we modified it to have restricted size of signatures. And our modification does not affect the security of it. Proof of security of $\sum_{\mathcal{P}}$, which is shown in [**?**] is slightly changed and it is obvious as we shall see later. The authentication trees will be the binary trees. Although the smart card forget the passed authentication values periodically, the simulation in the proof will not forget the values and the proof of $\sum_{\mathcal{P}}$ [**?**] will be applied to the trees.

Secondly, Schnorr's preprocessing is secure even for $k = 8$ in our application. In the de Rooij's attsck two equations from two signatures, (y_{i_0}, e_{i_0}) and (y_{i_1}, e_{i_1}), have the same coefficients in r_i $(0 \le i \le k-2)$ if and only if $i_0 \equiv i_1 \bmod k-1$ and the corresponding permutations $a_{i_0}(\cdot)$ and $a_{i_1}(\cdot)$, are identical[**?**]. Thus, if one possesses the signatures (y_i, e_i) for $i_0 < i \le i_1 + 1$ and if $i_0 \equiv i_1 \pmod{k-1}$, one has a candidate to determine the secret key s. But in the application of it to our scheme, the determination of $w_1, \cdots, w_d$ requires more steps in addition. Determining d unknowns requires d linear equations, so it will require $(k!)^{2d}$ steps. Hence if we let $d = 3, k = 8$, then the security is about 2^{93}, which is larger than the intended security 2^{72} of Schnorr. So our scheme in this paper has sufficient security.

Proof of the security of the modified scheme

Cramer-Damgård have shown that interactive protocols $\mathcal{P}$'s of certain types can be transformed into secure protocols $\sum_{\mathcal{P}}$'s. Our scheme has the interactive protocol $\mathcal{P}$ in the Figure 3.1 as a primitive, and $\mathcal{P}$ is considered collision intractible in the sense that there is no probabilistic polynomial time algorithm that, given X as input, can generate two accepting conversations (with respect to X) $(A, c, r), (A, c', r')$, with $c \neq c'$, except with negligible probability of success.

We do not consider the Schnorr preprocessing here, so the following z_b^i's are chosen at random. And we need a special simulator $\mathcal{S}$ such that on input X and any challenge $c \in \{0,1\}^{dN}$, $\mathcal{S}$ outputs an accepting conversation (A, c, r) of $\mathcal{P}$.

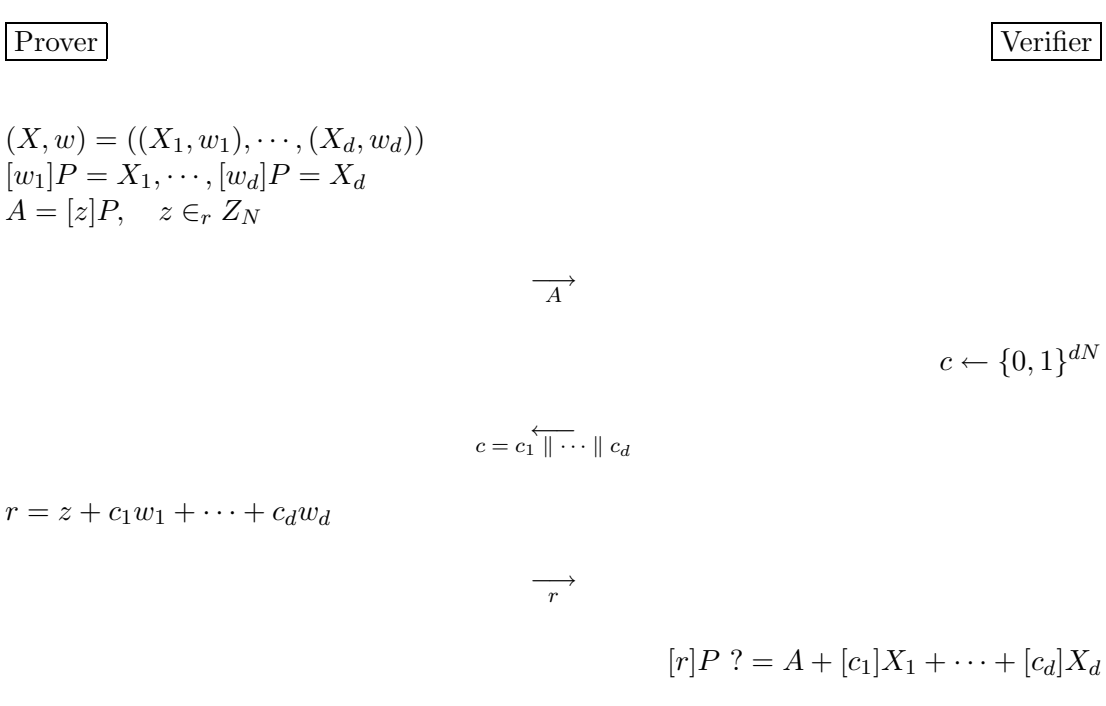

Figure 1: Protocol $\mathcal{P}$, common input $X = (X_1, \cdots, X_d)$

Theorem *Any probabilistic polynomial time crcking algorithm $\mathcal{A}$ that forges a signature on a new message with probability $\epsilon(k)$, after at most polynomially many calls to a signer, can be compiled into probabilistic polynomial time procedure $\mathcal{A}^*$ that, breaks the collision interactability of $\mathcal{P}$ with probability of the order of $\epsilon(k)$. The running time of $\mathcal{A}^*$ is of the same order as the running time of $\mathcal{A}$.*

Proof. Let X be an instance of $\mathcal{P}$. We now describe how $\mathcal{A}^*$ can cracks the collision intractability of $\mathcal{P}$ by using the forger $\mathcal{A}$ and the following simulation of $\sum_{\mathcal{P}}$. $\mathcal{A}^*$ receives X as input.

$\mathcal{A}^*$ first finds a solved instance $(X', w') = ((X_1, w_1), \cdots, (X_d, w_d))$. Then a bit e is chosen at random. Put $(X_e, w_e) = (X', w')$, and $X_{1-e} = X$.

For the simulation, we distinguish between two cases.

case $e = 0$: We create authentication trees with $P(k)(\leq (c+1)s)$ internal nodes, starting at the leaves. The leaves A_1^j are generated as follows.

1. $c^j \longleftarrow \{0,1\}^{dN}$
2. $(A_1^j, c^j, r_1^j) \longleftarrow \mathcal{S}(X_1, c^j)$.

For children $A_1^{p(2s+1)+2q}$ and $A_1^{p(2s+1)+2q+1}, 0 \leq p \leq c, 1 \leq q \leq s$, generate $A_0^i = [z_0^i]P$, with $i = ps + q, z_0^i \in_R Z_N$. Then the parent $A_1^{p(2s+1)+q}$ will be generated as

$$(A_1^{p(2s+1)+q}, A_1^{p(2s+1)+2q} \parallel A_1^{p(2s+1)+2q+1} \parallel A_0^i, r_1^i)$$
$$\longleftarrow \mathcal{S}(X_1, A_1^{p(2s+1)+2q} \parallel A_1^{p(2s+1)+2q+1} \parallel A_0^i).$$

In this way, the authentication trees are generated. If $c'(< c+1)$ number of authentication trees are generated, the other $c - c' + 1$ number of roots are chosen at random. The resulting instance $(X_0, X_1, A_1^1, A_1^{2s+2}, \cdots, A_1^{c(2s+1)+1})$ of $\sum_{\mathcal{P}}$ is sent to the forger $\mathcal{A}$. After this, the cracking algorithm can start making its (at most $P(k)$) calls. Note that this simulation can now deal with any signature request, as the i-th signature, on a message M^i, can be computed by computing $r_0^i = z_0^i + M_1^i w_1 + \cdots + M_d^i w_d^i$, with $M_1^i \parallel \cdots \parallel M_d^i \equiv \mathcal{H}(M^i) \bmod dN$.
case $e = 1$:

1. Generate $A_1^1 = [z_1^1]P, A_1^{2s+2} = [z_1^{2s+2}]P, \cdots, A_1^{c(2s+1)+1} = [z_1^{c(2s+1)+1}]P$ with $z_1^1, \cdots, z_1^{cs+1} \in_R Z_N$ and send the instance $(X_0, X_1, A_1^1, A_1^{2s+2}, \cdots, A_1^{c(2s+1)+1})$ to the forger $\mathcal{A}$.
2. Let $M^i \in \mathcal{M}$ be the i-th message to be signed. Generate $A_0^i \longleftarrow \mathcal{S}(X, \mathcal{H}(M^i))$. Proceed as the signature generation phase 3.3. Note again that z_b^i are chosen at random without considering the preprocessing.

Note that in both cases the simulation can deal with any signature request, by the special simtlator $\mathcal{S}$. Furthermore, the distribution of the A_0^i, r_0^i, A_1^i and r_1^i is always according to the honest signer who has access to both w_0 and w_1. Thus the simulation is perfect, and we may now assume that the cracking algorithm, outputs a forgery on a new message (i.e, a message that has not been signed by the simulatior) $\tilde{M}$. Without loss of generality, we assume that this happens after exactly $P(k)$ calls, with probability $\epsilon(k)$.

Let $(Auth(A_0), r_0)$ be the forgery, on a new message $\tilde{M}$. Suppose that $A_0 = A_0^j$ for some $1 \le j \le P(k)$, with probability $\epsilon_1(k)$. As $\tilde{M}$ has not been signed by the simulation, we must have $\tilde{M} \neq M^j$, so $\mathcal{A}^*$ can get a collision for $\mathcal{P}$ from $(A_0, \tilde{M}, r_0)$ and (A_0^j, M^j, r_0^j).

If, on the contrary, $A_0 \neq A_0^j$ for all $1 \le j \le P(k)$, then there clearly exists a tuple $(A_1', A_1'', A_1''', A_0', r_1')$ in $Auth(A_0)$ and a node A_1^i in the tree, with $A_1' = A_1^i$, such that A_1^i is a leaf or A_1^i is an internal node with $A_1'' \parallel A_1''' \parallel A_0' \neq A_1^{p(2s+1)+2q} \parallel A_1^{p(2s+1)+2q+1} \parallel A_0^i$, with $i = ps + q, 0 \le p \le c, 1 \le q \le s$.

In case A_1^i is an internal node, say with probability $\epsilon_2(k)$, we immediately get a collision. If A_1^i is a leaf, with probability $\epsilon_3(k)$, however, the probability that $A_1'' \parallel A_1''' \parallel A_0' \neq c^i$ is $1 - \frac{1}{2^{dN}}$, as the distribution of A_1^i is independent of the distribution of c^j (by the properties of the special simulator), and c^j was chosen uniformly at random. Thus in this case we get a collision with probability $1 - \frac{1}{2^{dN}}$. From the perfectness of the simulation it follows that the distribution of everything sent to $\mathcal{A}$ is independent of e. Therefore the probability that $\mathcal{A}^*$ can compute a collision for the instance $X_{1-e} = X$ is

$$\frac{1}{2}\epsilon_1(k) + \frac{1}{2}\epsilon_2(k) + \frac{1}{2}(1 - \frac{1}{2^{dN}})\epsilon_3(k) \ge \frac{1}{2}\epsilon(k) - \frac{1}{2^{dN+1}}\epsilon_3(k),$$

which is clearly of the same order as $\epsilon(k)$. Thus we have shown that any forger of the signature scheme $\sum_P$ can be turned very efficiently into a cracker of the collision intractibility of $\mathcal{P}$, with essentially the same probability of success.

3.6 Conclusion

We modified the original scheme of Cramer-Damgård[**?**] for smart card application. The use of elliptic curve cryptosystem and Schnorr's preprocessing with the periodic renewal of authentication is crucial for smart card application. The computational effort in the smart card is mainly due to the computations for $A_b^{v^*}$'s in the preprocessing, which are additions and doublings. So the use of elliptic curves over F_2 will be better than over other base fields for hardware implementation. Our suggested signature scheme has bounded size of signature and the computation in the smart card is very effective with high security.

References

[1] S. Goldwasser, S. Micali and R. Rivest, *A Digital Signature Scheme Secure Against Chosen Message Attacks*, SIAM Journal on Computing, 17(2):281-308, 1988.

[2] C. P. Schnorr, *Efficient Identification and Signatures for Smart Cards*, Advances in Cryptology-Crypto'89, LNCS 435, Springer-Verlag, 1990, pp.239-252.

[3] C. P. Schnorr, *Efficient Signature Generation by Smart Cards*, J. Cryptology, 4(3), 1991, pp.161-174.

[4] P. de Rooij, *On the Security of the Schnorr Scheme Using Preprocessing*, Advances in Cryptology-EUROCRYPT'91, LNCS 547, Springer-Verlag, 1991, pp.71-78.

[5] P. de Rooij, *On Schnorr's Preprocessing for Digital Signature Schemes*, Advances in Cryptology-EUROCRYPT'93, LNCS 765, Springer-Verlag, 1994, pp.435-439.

[6] R. Cramer, I. Damgård, *Secure Signature Schemes Based on Interactive Protocols*, Advances in Cryptology-Crypto'95, LNCS, Springer-Verlag, 1995, pp.297-310.

[7] A. J. Menezes, *Elliptic Curve Public Key Cryptosystems*, Kluwer Academic Publishers, 1993.

[8] A. J. Menezes, T. Okamoto, and S. A. Vanstone, *Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field*, Proc. 23rd, ACM Symp. Theory of Computing, 1991.

[9] W. Meier, O. Staffelbach, *Efficient Multiplication on Certain Non-supersingular Elliptic Curves*, Advances in Cryptology-Crypto'92, LNCS, Springer-Verlag, 1993, pp.333-344.

[10] J. H. Silverman, *The Arithmetic of Elliptic Curves*, Springer-Verlag, 1992.

[11] J. Solinas, *An Improved Algorithm for Arithmetic on a Family of Elliptic Curves*, Advances in Cryptology-Crypto'97, LNCS, Springer-Verlag, 1997, pp.357-371.

[12] R. Cramer, I. Damgàrd, *New Generation of Secure and Practical RSA-based Signatures*, Advances in Cryptology-Crypto'96, LNCS, Springer-Verlag, 1996, pp.173-185.

[13] C. Dwork, M. Naor, *An Efficient Existentially Unforgeable Signature Scheme and its Applications*, Advances in Cryptology-Crypto'94, LNCS 839, Spriner-Verlag, 1994, pp.234-246.

[14] D. Pointcheval, J. Stern, *Security Proofs for Signature Schemes*, Advances in Cryptology-EUROCRYPT'96, LNCS 1070, Springer-Verlag, pp.387-398.

[15] N. Koblitz, *A Course in Number Theory and Cryptography*, Springer-Verlag, 1987.