Meta-ElGamal signature schemes

Patrick Horster · Holger Petersen · Markus Michels

Theoretical Computer Science and Information Security,
University of Technology Chemnitz-Zwickau,
Straße der Nationen 62, D-09111 Chemnitz, Germany
Email: {pho,hpe,mmi}@informatik.tu-chemnitz.de

May 31, 1994

Abstract

There have been many approaches in the past to generalize the ElGamal signature scheme. In this paper we integrate all these approaches in a Meta-ElGamal signature scheme. We also investigate some new types of variations, that haven't been considered before. By this method we obtain in our example settings numerous variants of the ElGamal scheme. From these variants, we can extract new, highly efficient signature schemes, which haven't been proposed before. As an example, we present efficient DSA-variants.

1 Introduction

There have been many approaches to generalize the ElGamal signature scheme [ElGa84, AgMV90, NIST91, YeLa93, Knob93, Harn94, NyR293, NyRu94, HoP194, HoP294, HoP394]. In this paper we try to integrate all these approaches in a Meta-ElGamal signature scheme. We also investigate some new types of variation, that haven't been considered before. By this method we obtain more than 13.000 variants of the ElGamal scheme. The advantage of this proceeding is that we can extract highly efficient schemes out of these variants. Furthermore some of the proposed variants can be used for schemes of other concepts, e.g. blind signatures and multisignatures (see [HMP194, HMP294, HMP394] for detail), while the original ElGamal digital signature can't be used in these schemes.

CSS '94– 11/94 Fairfax Va., USA
©1994 ACM 0–89791–732–4/94/0011..$3.50

In section two we develop a Meta-ElGamal signature scheme for one message block. This scheme is extended to a Meta-scheme for two message blocks in section three and to a Meta-scheme for three message blocks in section four. This type of scheme hasn't been mentioned before. From all three schemes we define a "Mega"-ElGamal signature scheme in section five which includes most of the proposed schemes. We don't consider message recovery schemes as described in [NyR193, NyR293, NyRu94] as generalizations of the ElGamal scheme here, because the message is not transmitted explicitely and thus these schemes can't be used in text hashing mode. In section six we propose efficient variants for the DSA-algorithm and give a proposal, how the DSA-algorithm should have been defined. Finally, we give a conclusion and discuss some remarkable applications for the new schemes.

2 The Meta-ElGamal scheme for one message block

The basic scheme The certification authority chooses a large prime p and a generator α in $\mathbf{Z}_p$ and publishes p and α.

For an ElGamal signature [ElGa84] the signer *Alice* chooses a random number $x_A \in \mathbf{Z}_{p-1}$ and computes $y_A := \alpha^{x_A} \pmod p$. She publishes y_A as public and keeps x_A secret. These values are constant for all messages to be signed. To sign a message $m \in \mathbf{Z}_{p-1}$ Alice chooses a random number $k \in \mathbf{Z}^*_{p-1}$. She computes $r := \alpha^k \pmod p$ and solves the congruence

$$m \equiv x_A r + ks \pmod{p-1} \qquad (1)$$

for the parameter s. The triple (m, r, s) is the signed message. It can be verified by checking the congruence

$$\alpha^m \equiv y_A^r r^s \pmod p. \qquad (2)$$

First generalization (Mode of operation) To reduce the length of the signature it's possible to compute the parameters modulo a number q which is a divisor of $p-1$. Schnorr proposed to choose α as a generator of a multiplicative subgroup of $\mathbf{Z}_p$ of order $q \in \mathbf{P}$ and to calculate the parameter s modulo q [Schn89]. Later the National Institute for Standardization proposed to reduce additionally the parameter r modulo q [NIST91] and to solve the equation (1) for the parameter s, such that the verification can be done by the equation

$$r \pmod q \equiv \left(\alpha^{ms^{-1}} y_A^{-r \pmod q)s^{-1}} \pmod p\right) \pmod q \qquad (3)$$

A generalization of this reduction was proposed by Knobloch [Knob93], who suggested to reduce s modulo q and the parameter $r := \alpha^k \pmod p$ by a function $h : \mathbf{Z}_p \rightarrow \mathbf{Z}_{q_1}$ where q_1 is any number in the range between $2^{50} \leq q_1 \leq 2^{160}$. The function h can be a hash function, a one-way function or even a selection of $|q_1|$ bit out of the binary representation of $r \in \mathbf{Z}_p$. The generalized verification equation can be written as

$$h(r) \equiv h\left(\alpha^{ms^{-1}} y_A^{-h(r)s^{-1}} \pmod p\right) \qquad (4)$$

We can summarize these four possible choices of the parameters r, s, p and q as the possible modes of operation in the following table. We define the signature of m to be $(h(r), s)$.

Mode	$\|p\|$	$\|q\|$	$\|q_1\|$	$h(r)$	r	s
XL	512	512	512	*id*	$\mathbf{Z}_p$	$\mathbf{Z}_{p-1}$
L	512	160	512	*id*	$\mathbf{Z}_p$	$\mathbf{Z}_q$
M	512	160	160	*id*	$\mathbf{Z}_q$	$\mathbf{Z}_q$
S	512	160	50–160	any	$\mathbf{Z}_{q_1}$	$\mathbf{Z}_q$

The mode is characterized by the choices of $|q|$ and $|q_1|$ which will be considered in the following Meta-scheme. Of course, it's possible to change the size of the numbers p, q and q_1 in the future, if any improvements on calculating the discrete logarithm problem are made. The size of q and q_1 are strongly related to the size of p by the desired security, so that only the size of p will be considered in the Meta-scheme as a security parameter.

Additionally it seems possible to compute the signatures modulo a composite modul $n = p_1 \cdot p_2 \cdots p_l$, $p_i \in \mathbf{P}$. This mode will be termed as *Mode*=C. Note that there are security problems if the security relays on the factorization problem only. A detailed reflection is omitted here.

Second generalization (Number of variant) This generalization is based on the idea of Agnew, Mullin and Vanstone [AgMV90]. Instead of signature generation by the equation (1) we can also choose the general equation

$$A \equiv x_A B + kC \pmod q \qquad (5)$$

and choose A, B, C as a permutation of the parameters (m, r, s). The signature verification can be done by verifying the equation

$$\alpha^A \equiv y_A^B r^C \pmod p \qquad (6)$$

This generalization was published in [NyR293, NyRu94] and independently in [HoP194, HoP294]. By this method we obtain six possible signature schemes to which we will refer to with *No* $1, 2, 3, 4, 5, 6$. The following tabular lists all variants and the corresponding signature and verification equations.

No	A	B	C	signature (5)	verification (6)
1	m	r	s	$m \equiv x_A r + ks$	$\alpha^m \equiv y_A^r r^s$
2	m	s	r	$m \equiv x_A s + kr$	$\alpha^m \equiv y_A^s r^r$
3	s	r	m	$s \equiv x_A r + km$	$\alpha^s \equiv y_A^r r^m$
4	s	m	r	$s \equiv x_A m + kr$	$\alpha^s \equiv y_A^m r^r$
5	r	s	m	$r \equiv x_A s + km$	$\alpha^r \equiv y_A^s r^m$
6	r	m	s	$r \equiv x_A m + ks$	$\alpha^r \equiv y_A^m r^s$

Third generalization (Type of equation) This generalization has already been mentioned for one variant in ElGamal's Ph.D. thesis [ElGa84]. Instead of choosing the parameters A, B, C as a permutation of (m, r, s) we can also choose them as a permutation of $(mr, s, 1)$, $(mr, ms, 1)$, $(mr, rs, 1)$ and $(ms, rs, 1)$. There are two more possible permutations, namely $(ms, r, 1)$ and $(m, rs, 1)$, which we will call $ms-$ and $rs-$variant but they are not secure against forgery[1].

Summarizing the above results, we get five possible signature types, which we will refer to with the abbreviation EG for ElGamal followed by capital roman numbers I, II, III, IV, V. The first two have already been proposed in [NyR293, HoP194]. Some special variants have also been independently published in [YeLa93, Harn94]. The following table gives an complete overview about these variants:

Type	$(\pm A, \pm B, \pm C)$ permutation of		
EG I	m	r	s
EG II	mr	s	1
EG III	mr	ms	1
EG IV	ms	rs	1
EG V	mr	rs	1

The types EG III, EG IV and EG V are equivalent to *Type* EG I. We can see this by multiplying m^{-1} in *Type* EG III, s^{-1} in *Type* EG IV and r^{-1} in *Type* EG V. It makes sense to distinguish them from the others as we will see later.

Fourth generalization (Permutation of signs) Instead of choosing one of the five permutations for A, B, C we can also multiply each component by ± 1,

[1] The insecurity of the $rs-$variant was pointed out by Knobloch [Knob94].

such that $\pm A, \pm B, \pm C$ is one of the proposed permutations. By this method we get four times more variants, which we will refer to as $\sigma(1), \sigma(2), \sigma(3), \sigma(4)$ with $\sigma(i) = \langle \sigma_A(i), \sigma_B(i), \sigma_C(i) \rangle$. The signature equation can be written as

$$\sigma_A(i)A \equiv \sigma_B(i)Bx_A + \sigma_C(i)Ck \pmod q. \qquad (7)$$

σ is defined as in the following tabular:

i	σ	$\sigma_A(i)$	$\sigma_B(i)$	$\sigma_C(i)$
1	$\sigma(1)$	$+1$	$+1$	$+1$
2	$\sigma(2)$	$+1$	$+1$	-1
3	$\sigma(3)$	$+1$	-1	$+1$
4	$\sigma(4)$	$+1$	-1	-1

There are four more permutations of the signs, which can be obtained by multiplying the signature equation by -1, thus they are equivalent to the above choices and are omitted here.

In practice we would choose σ such that Alice needs no multiplication by -1 to calculate the signature parameter s. We will hence write the equation (7) less formal as

$$\pm A \equiv \pm Bx_A \pm Ck \pmod q. \qquad (8)$$

Fifth generalization (Choice of functions f and g) This generalization hasn't been proposed as yet. It's based on the idea of the variant in [Harn94]. In the types EG II – EG V there are at least once two of the parameters n, r, s which are multiplied. Instead of multiplication, we can also take operations $f, g : \mathbf{Z}_q{}^2 \rightarrow \mathbf{Z}_q$ which must be at least invertible in one of the two arguments. In the following, we consider them to be also commutative and invertible in both arguments, that is

- for all $a, b \in \mathbf{Z}_q : f(a, b) = f(b, a)$,
- for all $a, b \in \mathbf{Z}_q : f(a, b) = z \Leftrightarrow f^{-1}(z, a) = b$ and $f^{-1}(z, b) = a$, that is
  $f^{-1}(f(a, b), a) = b$ and $f^{-1}(f(a, b), b) = a$.

Let $\pm A, \pm B, \pm C$ be one of the following permutations:

Type	$(\pm A, \pm B, \pm C)$ permutation of		
EG I	m	r	s
EG II	$f(m, r)$	s	1
EG III	$f(m, r)$	$g(m, s)$	1
EG IV	$f(m, s)$	$g(r, s)$	1
EG V	$f(m, r)$	$g(r, s)$	1

In general the operations f and g are non-commutative functions and invertible (in polynomial time) in one argument. This gives a multitude of additional types of equations. If we look for instance at *Type* EG III, we would get as permutations EG IIIa: $(f(m, r), g(m, s), 1)$, EG IIIb: $(f(r, m), g(m, s), 1)$, EG IIIc: $(f(m, r), g(s, m), 1)$, EG IIId: $(f(r, m), g(s, m), 1)$. In *Type* EG IIIa and EG IIIb the operation g should be invertible in the first argument and in *Type* EG IIIc and EG IIId it should be invertible in the second argument to solve the signature equation for the parameter s.

In *Type* EG V we have to choose the operations f and g carefully, such that the equation can be solved for the parameter s. This is not the case for all imaginable choices of f and g.

For example, we could additionally choose the addition, the bitwise XOR and the bitwise Equivalent($\leftrightarrow$) such that $f, g \in \{\cdot, +, \oplus, \leftrightarrow\}$.

We will refer to the operations f and g by the tuple (f, g).

Sixth generalization (Choice of general functions) To complete the generalizations, we can adapt the ideas of the fifth generalization by writing the coefficients A, B, C as permutations of general functions e, f and g with arguments m, r and s. This leads us to further variants, which have some drawbacks compared to the variants proposed so far. The most important conditions on the functions $e, f, g : \mathbf{Z}_q{}^3 \rightarrow \mathbf{Z}_q$ and their arguments $m, r, s \in \mathbf{Z}_q$ are the following:

- To guarantee the solvability of the new signature equation it seems necessary that the parameter s is either only used in one of the three functions e, f, g or, in case of multiple occurrence of s, the functions have to be chosen such, that s can be extracted.
- To guarantee the security of the new signature schemes, each parameter m, r, s has to occur as argument in at least one of the three functions e, f, g.
- If two or three functions use exactly the same arguments, then they have to be chosen as different operations to avoid substitutional attacks and universal forgery.
- If m, s occur as arguments in one of the functions, then either m or s has to appear in one of the other two functions, to avoid the construction of an insecure $ms-$variant.
- If r, s occur as arguments in one of the functions, then either r or s has to appear in one of the other functions, to avoid the appearance of the insecure $rs-$variant.
- To get variants with efficient signature generation, we should choose the operations used together with the parameter s unequal to any operation with unefficient calculation of inverses.

- To obtain variants with efficient signature verification, we should choose one of the functions e, f, g equal to 1 to result in two exponentiations for signature verification.

To construct a new variant, at least the first five conditions should be noticed. Considering all conditions except the last one, we get for example an interesting variant, namely to choose A, B, C as a permutation of $e(m,r,s), f(m,r), g(m,r)$, with the function e unequal to multiplication, and different functions f and g. If one of the two functions f, g depends on only one argument, the other should be chosen unequal to multiplication, to avoid the occurrence of a variant, which is equivalent to one of the two insecure $rs-$ or $ms-$variants.

As the signs of the parameters A, B, C are included in the general functions e, f, g, we don't have to consider them any more in the general case. Additionally, every parameter that depends on only one argument $u \in \{m, r, s\}$, can be written more general as $f(u)$, where f can be any suitable function like multiplication by -1, bitwise inversion, reduction modulo a suitable modulus or others.

For example, in variant EG I.1 the parameters A, B, C can be chosen as a permutation of $(e(m), f(r), g(s))$. If the binary weight of the parameter r is bigger than $|p|/2$, we can choose the function f as bitwise inversion, to get a more efficient variant for signature verification. Or we can choose g as inversion (that is $g(s) = s^{-1} \pmod q$) to get a variant with only two exponentations for signature verification like in variant EG IV.1.

This sixth generalization is also part of the Meta-scheme, but it is omitted in the following description, because we get a total of 304 variants, described by different functions e, f, g and the possible permutations, which comply with the above conditions. In addition, the most important variants regarding the security and the efficiency already occurred in the first five generalizations.

Seventh generalization (Calculation of signature parameter r) Instead of choosing a random number $k \in \mathbf{Z}_q$ and computing $r := \alpha^k \pmod p$ we can also compute $r' := \alpha^k \pmod p$ and $r := d(r', m)$, where $d : {\mathbf{Z}_p}^2 \to \mathbf{Z}_p$ is a suitable function. The rest of the scheme is the same, except, that the message m doesn't need to occur in the functions e, f and g. This gives us additionally a total of 304 variants, in which we use the same choices for the functions e, f and g, another 40 variants, in which the parameter m doesn't appear in e, f and g, and 18 variants, in which the condition from the sixth generalization to avoid the $ms-variants$ are omitted. For example we can choose the function $d(r', m) := (r')^{-1} m \pmod p$ like in the Nyberg-Rueppel signature scheme giving message recovery [NyR193] or we can choose $d(r', m) := (r')^m = \alpha^{km} \pmod p$. The verification equation is the following:

$$r \equiv d\left(\alpha^{AC^{-1}} y_A^{-BC^{-1}} \pmod p, m\right). \tag{9}$$

This is different from equation (6), because α^k can't be computed directly, if d is not invertible in the first argument. Otherwise this can be done by the equation $\alpha^k \pmod p = r' = d^{-1}(r, m)$.

Note, that in this paper the message m is transmitted in every case, even if the function d was chosen suitable, such that message recovery would be possible. This allows us to use our variants in the text-hashing mode, which was also proposed in [NyR293, NyRu94] for the reduced MR(p)-ElGamal* message recovery scheme.

The other case, in which the message m is recovered can be found in detail in [HoPM94].

The Meta-ElGamal scheme Combining all generalizations except the sixth one, we get the Meta-ElGamal scheme (MEG) for one message block. The chosen signature scheme can be described by the following tuple:

$$MEG_1 = (Mode.Type.No.\sigma, d, f, g).$$

The parameters have the following meaning:

- $Mode \in$ {XL, L, M, S} gives the mode of operation,
- $Type \in$ {EG I, EG II, EG III, EG IV, EG V} gives the type of permutation,
- $No \in \{1,2,3,4,5,6\}$ gives the number of the permutation,
- $\sigma \in \{\sigma(1), \sigma(2), \sigma(3), \sigma(4)\}$ fixes the possible signs,
- $d : {\mathbf{Z}_p}^2 \to \mathbf{Z}_p$ specifies the computation of r,
- $f, g : {\mathbf{Z}_q}^2 \to \mathbf{Z}_q$ invertible in at least one argument.

For example the variant (XL.EG II.3.$\sigma(2), r', +, id$) describes the signature equation

$$s \equiv x_A(m + r) - k \pmod{p-1}$$

which has been proposed by Harn [Harn94]. Table 1 gives a complete survey of all possible signature equations for all types of permutation, without specializing the operations d, f, g and the mode of operation. With our example choices of f and g we obtain a total of 5088 ElGamal variants for one signature block.

No.	$\pm A$	$\pm B$	$\pm C$	signature	verification	Reference
EG I.1	m	r	s	$m \equiv x_A r + ks$	$\alpha^m \equiv y_A^r r^s$	[ElGa84]
EG I.2	m	s	r	$m \equiv x_A s + kr$	$\alpha^m \equiv y_A^s r^r$	[AgMV90]
EG I.3	s	r	m	$s \equiv x_A r + km$	$\alpha^s \equiv y_A^r r^m$	[HoP194]
EG I.4	s	m	r	$s \equiv x_A m + kr$	$\alpha^s \equiv y_A^m r^r$	[HoP194]
EG I.5	r	s	m	$r \equiv x_A s + km$	$\alpha^r \equiv y_A^s r^m$	[HoP194]
EG I.6	r	m	s	$r \equiv x_A m + ks$	$\alpha^r \equiv y_A^m r^s$	[HoP194]
EG II.1	1	$f(m,r)$	s	$1 \equiv x_A f(m,r) + ks$	$\alpha \equiv y_A^{f(r,m)} r^s$	[ElGa84]
EG II.2	1	s	$f(m,r)$	$1 \equiv x_A s + kf(m,r)$	$\alpha \equiv y_A^s r^{f(m,r)}$	[HoP194]
EG II.3	s	$f(m,r)$	1	$s \equiv x_A f(m,r) + k$	$\alpha^s \equiv y_A^{f(m,r)} r$	[Harn94]
EG II.4	s	1	$f(m,r)$	$s \equiv x_A + kf(m,r)$	$\alpha^s \equiv y_A r^{f(m,r)}$	[YeLa93]
EG II.5	$f(m,r)$	s	1	$f(m,r) \equiv x_A s + k$	$\alpha^{f(m,r)} \equiv y_A^s r$	[NyR293]
EG II.6	$f(m,r)$	1	s	$f(m,r) \equiv x_A + ks$	$\alpha^{f(m,r)} \equiv y_A r^s$	[NyR293]
EG III.1	1	$f(m,r)$	$g(m,s)$	$1 \equiv x_A f(m,r) + kg(m,s)$	$\alpha \equiv y_A^{f(m,r)} r^{g(m,s)}$	
EG III.2	1	$g(m,s)$	$f(m,r)$	$1 \equiv x_A g(m,s) + kf(m,r)$	$\alpha \equiv y_A^{g(m,s)} r^{f(m,r)}$	
EG III.3	$g(m,s)$	$f(m,r)$	1	$g(m,s) \equiv x_A f(m,r) + k$	$\alpha^{g(m,s)} \equiv y_A^{f(m,r)} r$	
EG III.4	$g(m,s)$	1	$f(m,r)$	$g(m,s) \equiv x_A + kf(m,r)$	$\alpha^{g(m,s)} \equiv y_A r^{f(m,r)}$	
EG III.5	$f(m,r)$	$g(m,s)$	1	$f(m,r) \equiv x_A g(m,s) + k$	$\alpha^{f(m,r)} \equiv y_A^{g(m,s)} r$	
EG III.6	$f(m,r)$	1	$g(m,s)$	$f(m,r) \equiv x_A + kg(m,s)$	$\alpha^{f(m,r)} \equiv y_A r^{g(m,s)}$	
EG IV.1	$f(m,s)$	$g(r,s)$	1	$f(m,s) \equiv x_A g(r,s) + k$	$\alpha^{f(m,s)} \equiv y_A^{g(r,s)} r$	
EG IV.2	$f(m,s)$	1	$g(r,s)$	$f(m,s) \equiv x_A + kg(r,s)$	$\alpha^{f(m,s)} \equiv y_A r^{g(r,s)}$	
EG IV.3	1	$g(r,s)$	$f(m,s)$	$1 \equiv x_A g(r,s) + kf(m,s)$	$\alpha \equiv y_A^{g(r,s)} r^{f(m,s)}$	
EG IV.4	1	$f(m,s)$	$g(r,s)$	$1 \equiv x_A f(m,s) + kg(r,s)$	$\alpha \equiv y_A^{f(m,s)} r^{g(r,s)}$	
EG IV.5	$g(r,s)$	1	$f(m,s)$	$g(r,s) \equiv x_A + kf(m,s)$	$\alpha^{g(r,s)} \equiv y_A r^{f(m,s)}$	
EG IV.6	$g(r,s)$	$f(m,s)$	1	$g(r,s) \equiv x_A f(m,s) + k$	$\alpha^{g(r,s)} \equiv y_A^{f(m,s)} r$	
EG V.1	$f(m,r)$	1	$g(r,s)$	$f(m,r) \equiv x_A + kg(r,s)$	$\alpha^{f(m,r)} \equiv y_A r^{g(r,s)}$	
EG V.2	$f(m,r)$	$g(r,s)$	1	$f(m,r) \equiv x_A g(r,s) + k$	$\alpha^{f(m,r)} \equiv y_A^{g(r,s)} r$	
EG V.3	$g(r,s)$	1	$f(m,r)$	$g(r,s) \equiv x_A + kf(m,r)$	$\alpha^{g(r,s)} \equiv y_A r^{f(m,r)}$	
EG V.4	$g(r,s)$	$f(m,r)$	1	$g(r,s) \equiv x_A f(m,r) + k$	$\alpha^{g(r,s)} \equiv y_A^{f(m,r)} r$	
EG V.5	1	$g(r,s)$	$f(m,r)$	$1 \equiv x_A g(r,s) + kf(m,r)$	$\alpha \equiv y_A^{g(r,s)} r^{f(m,r)}$	
EG V.6	1	$f(m,r)$	$g(r,s)$	$1 \equiv x_A f(m,r) + kg(r,s)$	$\alpha \equiv y_A^{f(m,r)} r^{g(r,s)}$	

Table 1: Meta-ElGamal signature schemes for one message block

Remark: If we consider the sixth generalization we can write the Meta-ElGamal sigature scheme as

$$MEG_1 = (Mode.Type.No, d, e, f, g)$$

with a suitable function $d : \mathbf{Z}_p{}^2 \to \mathbf{Z}_p$ and suitable functions $e,\ f,\ g : \mathbf{Z}_q{}^3 \to \mathbf{Z}_q$.

A concrete signature scheme For a concrete signature scheme, the public parameters used in the mode of operation have to be specified. For this reason, we first have to choose the size of the modul p as a security parameter. With this, the scheme can be described by the following parameters:

$$(MEG_1, |p|; p, q, q_1, \alpha, h).$$

The public parameters which are authentic to all users are

- two large primes $p, q \in \mathbf{P}$ with $q|(p-1)$,
- in *Mode* M and S a number q_1 ($2^{50} \leq q_1 \leq 2^{160}$) and a function $h : \mathbf{Z}_q \to \mathbf{Z}_{q_1}$,
- a generator $\alpha \in \mathbf{Z}_p$ of a cyclic subgroup of order q,

Signature generation:
Alice signs the message m by choosing a random number $k \in \mathbf{Z}_q$, computing $r' := \alpha^k \pmod p$, $r := d(r', m)$ and solving the equation

$$\pm A \equiv x_A \pm B \pm kC \pmod q. \qquad (10)$$

for the signature parameter s. This parameter can be written as

$$s := S_{MEG_1}(x_A, q, m, r, k).$$

Then she transmits the signed message (m, r, s) to the verifier Bob.

Signature verification:
Bob verifies the validity of the signature by checking the equation

$$r \equiv d\left(\alpha^{AC^{-1}} y_A^{-BC^{-1}} \pmod p), m\right). \qquad (11)$$

2.1 Security of the Meta-ElGamal scheme

In the following security analysis, we assume, that the used signature scheme is authentically known to the signer and all verifiers of the message. The other case, in which the scheme can be chosen at random for every message, will be discussed in section 5.1. By this assumption, the security analysis can be divided into three possible attacks [ElGa84]. In the first attack, we want to get the secret key of the signer. The second attack tries to forge a signature on any message m using all known public parameters. In the third attack the forger already knows a signature (r, s) on the message m and aims to forge another valid signature pair $(\tilde{r}, \tilde{s})$ on another message $\tilde{m}$ within the same signature scheme.

1. *Total break* (Computing the secret key x_A):
   As in the general signature equation (10) there are two unknown parameters, namely x_A and k, a forger can't get any information about x_A without knowing k. Thus the calculation of x_A seems to be as difficult as the computation of the discrete logarithm $log_\alpha(y_A)$.

2. *Universal forgery of messages:*
   To forge a message universally, the forger can first randomly choose an integer r, then attempts to find s or vice versa. In the first method, the forger must solve the discrete logarithm of

   - $\alpha^A y_A^{-B}$ in variants 1 and 6 for all types,
   - $\alpha^A r^{-C}$ in variants 2 and 5 for all types,
   - $y_A^B r^C$ in variants 3 and 4 for all types,

   In the second forgery, the attacker has the problem to solve the equation

   - $d \equiv e^r r \pmod p, d, e \in \mathbf{Z}_p$ in variants 1,3,5 and 6 for all types,
   - $d \equiv r^r \pmod p, d \in \mathbf{Z}_p$ in variants 2 and 4 for all types,

   These computations are extremely difficult and it's an open problem, if they are even more difficult than solving the discrete logarithm problem [AgMV90]. Of course if we can compute the discrete logarithm, we can apply one of the attacks above, so this question doesn't influence the security of the scheme.

   It seems possible, that equation (11) can be solved by simultaneous choice of r, s. In case of the $rs-$variants, this attack is indeed successful and results in a universal forgery of the signature scheme [Knob94]. For our Meta-scheme, it's still an open problem if there exists a similar attack. It's obvious, that in types EG IV and EG V, we can choose the arguments r, s in the operation g simultaneously and calculate the message m subsequently, but this gives only an existential forgery, which can be prevented as described below.

3. *Existential forgery of messages:*
   The existential forgery of messages is always possible as already pointed out in [ElGa84, ElGa85, HoP194, HoP294, Nybe94]. This substitutional attack is typically prevented by the use of a hash function [ElGa84, NIST91] or a redundancy generating function [NyR193, NyRu94]. In case of types EG I to EG V we have a much simpler attack than the one described in [ElGa84, ElGa85, Nybe94], which should hence be characterized for one variant of each type and can easily be applied to all of them:

   - In variant EG I.1, the attacker chooses random values $u, w \in \mathbf{Z}_q{}^*$, computes $r := \alpha^u y_A^{-w} \pmod p$, $s := rw^{-1} \pmod q$ and $m := us \pmod q$.
   - In variant EG II.1, the attacker chooses random values $u, w \in \mathbf{Z}_q{}^*$, computes $r := \alpha^u y_A^{-w} \pmod p$, $s := u^{-1} \pmod q$ and hence $m := f^{-1}(ws, r) \pmod q$.
   - In variant EG III.1, the attacker chooses random values $u, w \in \mathbf{Z}_q{}^*$ and computes $r := \alpha^u y_A^{-w} \pmod p$, $m := f^{-1}(wu^{-1}, r) \pmod q$ and $s := g^{-1}(u^{-1}, m) \pmod q$.
   - In variant EG IV.1 the attacker chooses random values $u, w \in \mathbf{Z}_q{}^*$ and computes $r := \alpha^u y_A^{-w} \pmod p$, $s := g^{-1}(w, r) \pmod q$ and $m := f^{-1}(u, s) \pmod q$.
   - In variant EG V.1 the attacker chooses random values $u, w \in \mathbf{Z}_q{}^*$ and computes $r := \alpha^u y_A^{-w} \pmod p$ $s := g^{-1}(w^{-1}, r) \pmod q$ and $m := f^{-1}(uw^{-1}, r) \pmod q$.

   The probability that the resulting m is a valid message, depends on the chosen redundancy scheme or the collision resistant hash function.

   Generally, this kind of existential forgery can be described in the following way: The attacker chooses $u, w \in \mathbf{Z}_q{}^*$ randomly and solves the equation $r := \alpha^u y_A^{-w} \pmod p$. Then he substitutes r with it's value in the equations $uC \equiv A \bmod q$ and $wC \equiv B \bmod q$ and tries to solve them for the signature parameters m and s. This is always possible in the *Types* EG I – EG V.

Equivalent security of the variants It is immediate, that some of the variants offer equivalent security [NyRu94]. For arbitrary choices of the operations f and g the equations

- 1 and 6,
- 2 and 4,
- 3 and 5,

for all types provide equivalent security, because only the roles of the generator α and the public key y_A are changed. Additionally, for special choices of the operations f and g, some of different types also provide equivalent security for randomly chosen messages (with no underlying redundancy scheme). For example, if we choose f, g as multiplication, then it is obvious, that all the variants EG I.j, EG II.j, EG III.j, EG IV.j and EG V.j ($j \in \{1, 2, 3, 4, 5, 6\}$ fixed) offer equivalent security, because the signed messages can be transformed into each other: If $(m; r, s)$ is a signed message in variant EG I.j, then

- $(m^{-1}; r, (m^{-1}s))$ is one in variant EG II.j,
- $(m^{-1}; r, s)$ is one in variant EG III.j,
- $(m; r, s^{-1})$ is one in variant EG IV.j and
- $(m; r^{-1}, -s)$ is one in variant EG V.j.

It is still an open problem, if such equivalences hold for all possible choices of f and g.

2.2 Performance of the Meta-ElGamal scheme

The most efficient schemes for the signer are those in which Alice doesn't need to compute any inverse modulo q for signature generation. In general these are the schemes j.3 and j.4 for $j \in \{$EG I, EG II, EG III, EG V$\}$. For $j \in \{$EG III, EG V$\}$ this is only true, if the operation g is not chosen as the multiplication. If we consider also the number of multiplications, then those schemes are most efficient, in which the operations f and g are chosen as bit operations, because these operations can be computed very efficiently. Furthermore the most efficient schemes for the verifier are those in which he needs two exponentiations instead of three. Thus the schemes j.3 and j.4 for $j \in \{$EG II, EG III, EG V$\}$ with a suitable function g for EG III and EG V seems to be best suited for digital signatures. Alice needs only one exponentiation for signature generation and Bob needs two exponentiations for signature verification.

3 Meta-ElGamal signatures for two message blocks

In his Ph.D. thesis, ElGamal also proposed to sign two message blocks m_1, m_2 with one signature (r, s) by solving the equation

$$m_1 \equiv x_A m_2 r + ks \pmod{p-1} \qquad (12)$$

for the parameter s. Applying the ideas of the seven generalizations except the sixth one described in the Meta-ElGamal scheme above, we get the following permutations for the parameters $\pm A, \pm B, \pm C$:

Type	$(\pm A, \pm B, \pm C)$ permutation of		
EG VI	$f(m_1, r)$	s	m_2
EG VII	$f(m_1, r)$	$g(m_2, s)$	m_2
EG VIII	$f(m_1, r)$	$g(m_1, s)$	m_2
EG IX	$f(m_1, s)$	$g(r, s)$	m_2
EG X	$f(m_1, r)$	$g(r, s)$	m_2

There are also two $ms-$ and one $rs-$variants in which $\pm A, \pm B, \pm C$ are permutations of $(f(m_1, s), m_2, r)$, $(f(m_1, s), g(m_2, r), 1)$ or $(m_1, m_2, f(r, s))$. These variants are not secure against forgery. The $ms-$ variants can also be examined if we append the hash value $h(m_1, m_2)$ of a suitable hash function to the message block m_2.

Table 2 gives a complete survey of all possible signature equations for all types of permutation without specializing the operations d, f, g and the mode of operation.

The definition of the Meta-ElGamal scheme is the same as in the case of one message block, that means that the generalized scheme can be written as

$$MEG_2 = (Mode.Type.No.\sigma, d, f, g).$$

The parameters have the same meaning as in chapter 2 with the slight difference that *Type* is chosen as

$$\mathit{Type} \in \{\text{EG VI, EG VII, EG VIII, EG IX, EG X}\}.$$

With our example choice of the operations f and g we get a total of 408 variants. For every variant, there exist four different choices of σ_i and four different choices of the mode of operation. This results in a total of 6528 ElGamal variants for two message blocks in our example without specialising the choice of the function d. It's also possible to apply the sixth generalization described in section 2, that is to choose A, B, C as a permutation of general functions $e, f, g : \mathbf{Z}_q{}^4 \rightarrow \mathbf{Z}_q$ with arguments m_1, m_2, r, s. Obviously, they have to fulfil several conditions to avoid forgery.

3.1 Security of the scheme

If we consider the chosen signature scheme to be authentically known to all participating parties, the security analysis for the total break and the universal forgery are the same as in the case of one message block. In the case of existential forgery, there is an obvious multiplicative attack for the choice of f and g as multiplication, which should be prevented by a suitable hash function or a redundancy generating function. If $(m_1, m_2; r, s)$ is a valid signed message, then

No.	$\pm A$	$\pm B$	$\pm C$	signature	verification
EG VI.1	m_2	$f(m_1,r)$	s	$m_2 \equiv x_A f(m_1,r) + ks$	$\alpha^{m_2} \equiv y_A^{f(m_1,r)} r^s$
EG VI.2	m_2	s	$f(m_1,r)$	$m_2 \equiv x_A s + k f(m_1,r)$	$\alpha^{m_2} \equiv y_A^s r^{f(m_1,r)}$
EG VI.3	s	$f(m_1,r)$	m_2	$s \equiv x_A f(m_1,r) + km_2$	$\alpha^s \equiv y_A^{f(m_1,r)} r^{m_2}$
EG VI.4	s	m_2	$f(m_1,r)$	$s \equiv x_A m_2 + k f(m_1,r)$	$\alpha^s \equiv y_A^{m_2} r^{f(m_1,r)}$
EG VI.5	$f(m_1,r)$	s	m_2	$f(m_1,r) \equiv x_A s + km_2$	$\alpha^{f(m_1,r)} \equiv y_A^s r^{m_2}$
EG VI.6	$f(m_1,r)$	m_2	s	$f(m_1,r) \equiv x_A m_2 + ks$	$\alpha^{f(m_1,r)} \equiv y_A^{m_2} r^s$
EG VII.1	m_2	$f(m_1,r)$	$g(m_2,s)$	$m_2 \equiv x_A f(m_1,r) + k g(m_2,s)$	$\alpha^{m_2} \equiv y_A^{f(m_1,r)} r^{g(m_2,s)}$
EG VII.2	m_2	$g(m_2,s)$	$f(m_1,r)$	$m_2 \equiv x_A g(m_2,s) + k f(m_1,r)$	$\alpha^{m_2} \equiv y_A^{g(m_2,s)} r^{f(m_1,r)}$
EG VII.3	$g(m_2,s)$	$f(m_1,r)$	m_2	$g(m_2,s) \equiv x_A f(m_1,r) + km_2$	$\alpha^{g(m_2,s)} \equiv y_A^{f(m_1,r)} r^{m_2}$
EG VII.4	$g(m_2,s)$	m_2	$f(m_1,r)$	$g(m_2,s) \equiv x_A m_2 + k f(m_1,r)$	$\alpha^{g(m_2,s)} \equiv y_A^{m_2} r^{f(m_1,r)}$
EG VII.5	$f(m_1,r)$	$g(m_2,s)$	m_2	$f(m_1,r) \equiv x_A g(m_2,s) + km_2$	$\alpha^{f(m_1,r)} \equiv y_A^{g(m_2,s)} r^{m_2}$
EG VII.6	$f(m_1,r)$	m_2	$g(m_2,s)$	$f(m_1,r) \equiv x_A m_2 + k g(m_2,s)$	$\alpha^{f(m_1,r)} \equiv y_A^{m_2} r^{g(m_2,s)}$
EG VIII.1	m_2	$f(m_1,r)$	$g(m_1,s)$	$m_2 \equiv x_A f(m_1,r) + k g(m_1,s)$	$\alpha^{m_2} \equiv y_A^{f(m_1,r)} r^{g(m_1,s)}$
EG VIII.2	m_2	$g(m_1,s)$	$f(m_1,r)$	$m_2 \equiv x_A g(m_1,s) + k f(m_1,r)$	$\alpha^{m_2} \equiv y_A^{g(m_1,s)} r^{f(m_1,r)}$
EG VIII.3	$g(m_1,s)$	$f(m_1,r)$	m_2	$g(m_1,s) \equiv x_A f(m_1,r) + km_2$	$\alpha^{g(m_1,s)} \equiv y_A^{f(m_1,r)} r^{m_2}$
EG VIII.4	$g(m_1,s)$	m_2	$f(m_1,r)$	$g(m_1,s) \equiv x_A m_2 + k f(m_1,r)$	$\alpha^{g(m_1,s)} \equiv y_A^{m_2} r^{f(m_1,r)}$
EG VIII.5	$f(m_1,r)$	$g(m_1,s)$	m_2	$f(m_1,r) \equiv x_A g(m_1,s) + km_2$	$\alpha^{f(m_1,r)} \equiv y_A^{g(m_1,s)} r^{m_2}$
EG VIII.6	$f(m_1,r)$	m_2	$g(m_1,s)$	$f(m_1,r) \equiv x_A m_2 + k g(m_1,s)$	$\alpha^{f(m_1,r)} \equiv y_A^{m_2} r^{g(m_1,s)}$
EG IX.1	$f(m_1,s)$	$g(r,s)$	m_2	$f(m_1,s) \equiv x_A g(r,s) + km_2$	$\alpha^{f(m_1,s)} \equiv y_A^{g(r,s)} r^{m_2}$
EG IX.2	$f(m_1,s)$	m_2	$g(r,s)$	$f(m_1,s) \equiv x_A m_2 + k g(r,s)$	$\alpha^{f(m_1,s)} \equiv y_A^{m_2} r^{g(r,s)}$
EG IX.3	m_2	$g(r,s)$	$f(m_1,s)$	$m_2 \equiv x_A g(r,s) + k f(m_1,s)$	$\alpha^{m_2} \equiv y_A^{g(r,s)} r^{f(m_1,s)}$
EG IX.4	m_2	$f(m_1,s)$	$g(r,s)$	$m_2 \equiv x_A f(m_1,s) + k g(r,s)$	$\alpha^{m_2} \equiv y_A^{f(m_1,s)} r^{g(r,s)}$
EG IX.5	$g(r,s)$	m_2	$f(m_1,s)$	$g(r,s) \equiv x_A m_2 + k f(m_1,s)$	$\alpha^{g(r,s)} \equiv y_A^{m_2} r^{f(m_1,s)}$
EG IX.6	$g(r,s)$	$f(m_1,s)$	m_2	$g(r,s) \equiv x_A f(m_1,s) + km_2$	$\alpha^{g(r,s)} \equiv y_A^{f(m_1,s)} r^{m_2}$
EG X.1	$f(m_1,r)$	m_2	$g(r,s)$	$f(m_1,r) \equiv x_A m_2 + k g(r,s)$	$\alpha^{f(m_1,r)} \equiv y_A^{m_2} r^{g(r,s)}$
EG X.2	$f(m_1,r)$	$g(r,s)$	m_2	$f(m_1,r) \equiv x_A g(r,s) + km_2$	$\alpha^{f(m_1,r)} \equiv y_A^{g(r,s)} r^{m_2}$
EG X.3	$g(r,s)$	m_2	$f(m_1,r)$	$g(r,s) \equiv x_A m_2 + k f(m_1,r)$	$\alpha^{g(r,s)} \equiv y_A^{m_2} r^{f(m_1,r)}$
EG X.4	$g(r,s)$	$f(m_1,r)$	m_2	$g(r,s) \equiv x_A f(m_1,r) + km_2$	$\alpha^{g(r,s)} \equiv y_A^{f(m_1,r)} r^{m_2}$
EG X.5	m_2	$g(r,s)$	$f(m_1,r)$	$m_2 \equiv x_A g(r,s) + k f(m_1,r)$	$\alpha^{m_2} \equiv y_A^{g(r,s)} r^{f(m_1,r)}$
EG X.6	m_2	$f(m_1,r)$	$g(r,s)$	$m_2 \equiv x_A f(m_1,r) + k g(r,s)$	$\alpha^{m_2} \equiv y_A^{f(m_1,r)} r^{g(r,s)}$

Table 2: The Meta-ElGamal-scheme for two message blocks

- $(z \cdot m_1, z \cdot m_2; r, z \cdot s)$ in *Type* EG VI and EG X,
- $(z \cdot m_1, z \cdot m_2; r, s)$ in *Type* EG VII and EG VIII,
- $(m_1, z \cdot m_2; r, z \cdot s)$ in *Type* EG IX,

are also valid signed messages of the same signer for random choices of $z \in \mathbf{Z}_q$. In addition, there is one simple attack of existential forgery for all types of equations which can be adapted from the described attack in section 2.1 by substituting m with m_1 and choosing additionally $m_2 \in \mathbf{Z}_q$ at random.

Equivalences to other variants It's obvious that some variants of the same type offer equivalent security like in the case of one message block (see section 2.1). Additionally, for special choices of the operations f and g and randomly chosen messages (with no underlying redundancy scheme) there exist equivalences to other types.

For example, if we choose f, g as multiplication, then all variants EG VI.j, EG VII.j, EG VIII.j, EG IX.j and EG X.j ($j \in \{1,2,3,4,5,6\}$ fixed) provide equivalent security, because the signed messages $(m_1, m_2; r, s)$ can be transformed into each other by linear transformations. Besides, we have equivalences to the signature schemes with one message block. For example, from a signed message $(m_1, m_2; r, s)$ in variant EG VI.j we can compute a valid signed message $(m_1^{-1} m_2; r, (m_1^{-1} s))$ in variant EG I.j and hence also valid signatures for all other types EG II.j, EG III.j, EG IV.j and EG V.j as already pointed out in section 2.1.

3.2 Performance of the scheme

The most efficient variants are those, in which Alice doesn't need to compute any inverse for signature generation. This is the case for the schemes $j.3$ and $j.4$ with $j \in$ {EG VI, EG VII, EG VIII, EG X} and the operation g unequal to the multiplication. If we are also counting the multiplications, those schemes are most efficient, in which the operations f, g are chosen as bit operations and not as arithmetic operations. In EG VI-X the verifier always needs three exponentiations.

4 Meta-ElGamal signatures for three message blocks

Obviously the ElGamal-scheme can also be used for signing three message blocks, if these blocks are within a special redundancy scheme. This gives us one more type of signature equation. The parameters $\pm A, \pm B, \pm C$ are a permutation of

$$(m_1, f(m_2, r), g(m_3, s)).$$

This will be refered to as *Type* EG XI. The chosen signature scheme can be described by the following tuple:

$$MEG_3 = (Mode.\text{EG XI}.No.\sigma, d, f, g).$$

In our example choice of the operations f and g we get a total of 1536 ElGamal variants for three message blocks without fixing the mode of operation and the choice of the function d. Table 3 gives a survey of all possible signature equations for all permutations of $\pm A, \pm B, \pm C$ without concrete choices of the operations f and g:

Obviously it's also possible to adapt the sixth generalization from section 2, that is to choose A, B, C as a permutation of functions $e, f, g : \mathbf{Z}_q{}^5 \rightarrow \mathbf{Z}_q$ with arguments m_1, m_2, m_3, r, s. It's an open problem to describe all necessary conditions for these functions to avoid any forgery to the resulting signature schemes.

4.1 Security of the scheme

If the chosen signature scheme is authentically known to all parties, then the security analysis for the total break and the universal forgery are the same as in the case of one and two message blocks. In the case of existential forgery, there are two more obvious attacks, which can be combined with each other:

1. For the choice of f and g as multiplication, there is a multiplicative attack, like in the case of two message blocks. If $(m_1, m_2, m_3; r, s)$ is a valid signed message, then

   - $(z \cdot m_1, z \cdot m_2, z \cdot m_3; r, s)$
   - $(z \cdot m_1, z \cdot m_2, m_3; r, z \cdot s)$

   are also valid signed messages for $z \in \mathbf{Z}_q$ by the same signer.

2. For the message m_3 there is also the same attack as for the $ms-$variant in the case of one message block. Given a valid signature (r, s) on the message (m_1, m_2, m_3) a forger can choose a message $(m_1, m_2, \tilde{m}_3)$ with random $\tilde{m}_3 \in \mathbf{Z}_q$ and compute the valid signature $(r, s \cdot m_3 \cdot \tilde{m}_3^{-1})$.

The first attack can be prevented by the choice of a suitable hash function or a redundancy generating function. The second attack is impossible, if we choose for example the last t bit of the message block m_2 to be the hash value $h(m_1, m_2', m_3)$ of a suitable hash function where m_2' denotes a $|p| - t$ bit message and $m_2 = m_2' || h(m_1, m_2', m_3)$. The parameter $t \in \mathbf{Z}_{p-1}$ denotes an additional security parameter.

Equivalences to other variants The equivalences between the variants of *Type* EG XI hold as in the case of one and two message blocks (see sections 2.1 and 3.1). Besides these, there are equivalences to signature schemes with two message blocks and hence to those with one message block. If we choose the operations f and g as multiplication, then from a given signed message $(m_1, m_2, m_3; r, s)$ in type EG XI.j we can compute a valid signed message $(m_1 m_3^{-1}, m_2 m_3^{-1}; r, s)$ in type EG VI.j $(j \in \{1, 2, \ldots, 6\})$. The other two obvious transformations by multiplication of m_1^{-1} and m_2^{-1} result in signatures in the insecure $ms-$variants.

4.2 Performance of the scheme

The most efficient variants are EG XI.3 and EG XI.4 if we choose the operation g not to be the multiplication. Alice needs one exponentiation for signature generation and the verifier Bob needs a total of three exponentiations for signature verification. Thus we need only one exponentiation per message block. The length of the signature depends on the chosen mode. If we take for example the DSA-mode, the signed message would contain $3|p| + 2|q|$ bit, which is an expansionrate of

$$\frac{3|p| + 2|q|}{3|p|} = 1 + \frac{2|q|}{3|p|}.$$

With $|p| = 512$ and $|q| = 160$ this value is equal to 1.208. In *Mode* S with $|q_1|$=64 the rate is only 1.145.

5 The Mega-ElGamal scheme

Combining the three proposed Meta-schemes for one, two and three message blocks, we get a "Mega"-scheme which can be written as

$$MEG = MEG_1 \cup MEG_2 \cup MEG_3$$

$$= (Mode.Type.No.\sigma, d, f, g).$$

The parameters can be chosen out of the following:

- *Mode* $\in$ {XL, L, M, S} gives the mode of operation,

No.	$\pm A$	$\pm B$	$\pm C$	signature	verification
EG XI.1	m_1	$f(m_2,r)$	$g(m_3,s)$	$m_1 \equiv x_A f(m_2,r) + kg(m_3,s)$	$\alpha^{m_1} \equiv y_A^{f(m_2,r)} r^{g(m_3,s)}$
EG XI.2	m_1	$g(m_3,s)$	$f(m_2,r)$	$m_1 \equiv x_A g(m_3,s) + kf(m_2,r)$	$\alpha^{m_1} \equiv y_A^{g(m_3,s)} r^{f(m_2,r)}$
EG XI.3	$g(m_3,s)$	$f(m_2,r)$	m_1	$g(m_3,s) \equiv x_A f(m_2,r) + km_1$	$\alpha^{g(m_3,s)} \equiv y_A^{f(m_2,r)} r^{m_1}$
EG XI.4	$g(m_3,s)$	m_1	$f(m_2,r)$	$g(m_3,s) \equiv x_A m_1 + kf(m_2,r)$	$\alpha^{g(m_3,s)} \equiv y_A^{m_1} r^{f(m_2,r)}$
EG XI.5	$f(m_2,r)$	$g(m_3,s)$	m_1	$f(m_2,r) \equiv x_A g(m_3,s) + km_1$	$\alpha^{f(m_2,r)} \equiv y_A^{g(m_3,s)} r^{m_1}$
EG XI.6	$f(m_2,r)$	m_1	$g(m_3,s)$	$f(m_2,r) \equiv x_A m_1 + kg(m_3,s)$	$\alpha^{f(m_2,r)} \equiv y_A^{m_1} r^{g(m_3,s)}$

Table 3: The Meta-ElGamal-scheme for three message blocks

- $Type \in \{$EG I, EG II, $\ldots$, EG XI$\}$ gives the type of permutation,
- $No \in \{1,2,3,4,5,6\}$ gives the number of the permutation,
- $\sigma \in \{\sigma(1), \sigma(2), \sigma(3), \sigma(4)\}$ fixes the possible signs,
- $d : \mathbf{Z}_p{}^2 \to \mathbf{Z}_p$ specifies the computation of r,
- $f, g : \mathbf{Z}_q{}^2 \to \mathbf{Z}_q$ invertible in at least one argument.

The general signature generation can be written as

$$sign_{MEG}(m_1, m_2, m_3) = (r, s)$$

without specializing the size of the chosen modul p, the concrete parameters p, q, α and in case of $Mode \in \{$M,S$\}$ the size of the parameter q_1, its value and the (hash-)function h. In the cases of $Type \in \{$EG I, EG II, EG III, EG IV, EG V$\}$ the messages m_2 and m_3 are chosen as $m_2 = m_3 = 1$, in case of $Type \in \{$EG VI, EG VII, EG VIII, EG IX, EG X$\}$ the message m_3 is equal to 1.

5.1 Security of the "Mega"-scheme

If we consider that the chosen signature scheme is authentically known to all parties the security of the "Mega"-scheme has already been analyzed in detail in the previous sections.

Here we assume, that the signature scheme can be chosen at random by the signer, without agreeing on it with the verifier before signature generation. This could be useful if we use the "Mega"-scheme e.g. for authenticated electronic mail, where the sender doesn't want to communicate with the recipient of the mail before to agree on a signature scheme or to send any additional authentic information together with his e-mail.

In this case, a forger could start a novel type of attack: If he already knows a valid signature (r, s) on the message m in one variant of the "Mega"-scheme he could try to forge a valid signature pair $(\tilde{r}, \tilde{s})$ for another message $\tilde{m}$ which is signed by the same user in a different variant of the "Mega"-scheme.

As we already mentioned in the security considerations above, for special choices of the operations f and g there exist equivalent variants from a security point of view, in which the signed messages can be transformed into each other, without regarding the underlying redundancy scheme of the messages. This leads to an attack, by choosing the new signed message as a transformation of a known signed message. To countermeasure this attack, we have to choose a suitable hash function h or a redundancy scheme R, such that for no two equivalent variants the two corresponding messages are within the same redundancy scheme.

For this purpose, assume that the tuple MEG can be coded as a l bit identification string ID_{Sig}. This identification string can be chosen as the first l bit of the message block $m_1 = ID_{Sig}||m_1'$ for all signature schemes. Additionally, in the case of one message block, the $|p|-l$ bit message m_1' should be a hash value or $m_1' \in R$. In case of two message blocks, the last t bit of m_2 (where $t \in \mathbf{Z}_{p-1}$ is a security parameter) can be chosen as the hash value $h(m_1, m_2')$ where m_2' is a $|p| - t$ bit message. In the case of three message blocks, the last t bit of m_2 can be chosen as the hash value $h(m_1, m_2', m_3)$.

6 Efficient variants for DSA

The DSA algorithm [NIST91] needs the computation of two inverses modulo q, namely k^{-1} and s^{-1}. To avoid at least one inversion, it's necessary, that either the parameter s or the parameter k is single in the signature equation.

6.1 Efficient variants for signature generation

The only equations, in which the parameter s is single are $j.3$ and $j.4$ for $j \in \{$EG I, EG II, EG VI$\}$. This gives a total of six efficient DSA-variants which are listed in table 4. The parameter A is always equal to s. Together with the choice of σ and the mode of operation S, we get a total of 48 DSA-variants with efficient signature generation, without specializing the operation f. The schemes $j.3, j.4$ with $j \in$ {EG III, EG V, EG VII, EG VIII, EG X, EG XI} and a suitable function g are further efficient variants.

6.2 Efficient variants for signature verification

If we are looking for the DSA verification equation (3), then the most efficient variants are those, in which the parameter k is already single in the equation. We can observe that this is the case for the variants EG II.3, EG II.5, EG III.3, EG III.5, EG IV.1, EG IV.6, EG V.2 and EG V.4 which leads us to a total of eight efficient DSA-variants which are given in table 5. The parameter C is always equal to 1. Together with the choice of σ and the mode of operation S, we get 64 DSA-variants with efficient verification, without specializing the choice of the operation f.

6.3 How the DSA should have been defined

The most efficient variant for both, the signer and the verifier is EG II.3, which needs no inversion for both parties. This scheme has been proposed as variant (XL.EG II.3.$\sigma(2), r', +, id$) in [Harn94] and as variants (L/M,EG II.3.$\sigma(2), r', \cdot, id$) in [NyR293, NyRu94].

The second best is the variant EG II.5, in which the inverse of x_A has to be computed only once for all signatures. For both equations, the choice of $f \in \{+, \oplus, \leftrightarrow\}$ would give an efficient variant, in which the signer has to compute one exponentiation modulo q and one multiplication. The verifier has to compute two exponentiations and one multiplication modulo q.

Other efficient variants with no inversion are the restricted variants EG III.3 and EG V.4 if we restrict the choice of the function g to additive or bitwise operations, which can be efficiently inverted, to solve the signature equation for the parameter s. With the same restriction for the function g, the variants EG III.5 and EG V.2 need only one inversion of the value x_A which can be precomputed.

7 Conclusion

We have presented Meta-ElGamal signature schemes for one, two and also for three message blocks which satisfy a given redundancy scheme. These Meta-schemes have been integrated in a so called parameterized "Mega"-scheme. Most of the previously known ElGamal-like signature schemes can be embedded in this approach. For further discussions, we only have to consider this general "Mega"-scheme and no longer the single signature schemes. Additionally we can extract very efficient variants from the "Mega"-scheme, which haven't been mentioned before as in the case of the proposed DSA-variants.

Furthermore there are interesting applications for weak and strong blind signature schemes [HMP194, HMP294, HoP394], for signature schemes giving message recovery [HoPM94, NyR293, NyRu94], for multisignatures [HMP394], for authentication protocols [HoP294] and for authentic key distribution and key agreement protocols [NyR293, HoP294] which can also be generalized applying the ideas of the "Mega"-ElGamal signature scheme.

References

[AgMV90] G.B.Agnew, R.C.Mullin, S.A.Vanstone, "Improved digital signature scheme based on discrete exponentiation", Electronics Letters, Vol. 26, (1990), pp. 1024–1025.

[ElGa84] T.ElGamal, "Cryptography and logarithms over finite fields", Stanford University, CA., UMI Order No. DA 8420519, (1984), 119 pages.

[ElGa85] T.ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms", IEEE Transactions on Information Theory, Vol. IT-30, No. 4, July, (1985), pp. 469–472.

[Harn94] L.Harn, "New digital signature scheme based on discrete logarithm", Electronics Letters, Vol. 30, No. 5, (1994), pp. 396 – 398.

[HMP194] P.Horster, M.Michels, H.Petersen, "Generalized blind signature schemes based on the discrete logarithm problem", (in preparation), (1994), 11 pages.[1]

[HMP294] P.Horster, M.Michels, H.Petersen, "Efficient blind signature schemes based on the discrete logarithm problem", (in preparation), (1994), 4 pages. [1]

[HMP394] P.Horster, M.Michels, H.Petersen, "Meta-Multisignature schemes based on the discrete logarithm problem", (in preparation), (1994), 11 pages.[1]

[HoP194] P.Horster, H.Petersen, "Generalized ElGamal-signatures" (in German), Sicherheit in Informationssystemen, Proceedings of SIS '94, Verlag der Fachvereine Zürich, (1994), pp. 89–106.

[HoP294] P.Horster, H.Petersen, "Signature and authentication schemes based on the discrete logarithm" (in German), Internal Report, RWTH Aachen, March, (1994), 96 pages.[1]

No.	$-B$	C	signature	verification
EG I.3	$h(r)$	m	$s \equiv -x_A h(r) + km$	$h(r) \equiv h(\alpha^{sm^{-1}} y_A^{h(r)m^{-1}})$
EG I.4	m	$h(r)$	$s \equiv -x_A m + kh(r)$	$h(r) \equiv h(\alpha^{sh(r)^{-1}} y_A^{mh(r)^{-1}})$
EG II.3	$f(m,h(r))$	1	$s \equiv -x_A f(m,h(r)) + k$	$h(r) \equiv h(\alpha^s y_A^{f(m,h(r))})$
EG II.4	1	$f(m,h(r))$	$s \equiv -x_A + kf(m,h(r))$	$h(r) \equiv h(\alpha^{sf(m,h(r))^{-1}} y_A^{f(m,h(r))^{-1}})$
EG VI.3	$f(m,h(r))$	m_2	$s \equiv -x_A f(m,h(r)) + km_2$	$h(r) \equiv h(\alpha^{sm_2^{-1}} y_A^{f(m,h(r))m_2^{-1}})$
EG VI.4	m_2	$f(m,h(r))$	$s \equiv -x_A m_2 + kf(m,h(r))$	$h(r) \equiv h(\alpha^{sf(m,h(r))^{-1}} y_A^{m_2 f(m,h(r))^{-1}})$

Table 4: Efficient DSA-variants for signature generation

No.	A	$-B$	signature	verification
EG II.3	s	$f(m,h(r))$	$s \equiv -x_A f(m,h(r)) + k$	$h(r) \equiv h(\alpha^s y_A^{f(m,h(r))})$
EG II.5	$f(m,h(r))$	s	$f(m,h(r)) \equiv -x_A s + k$	$h(r) \equiv h(\alpha^{f(m,h(r))} y_A^s)$
EG III.3	$g(m,s)$	$f(m,h(r))$	$g(m,s) \equiv -x_A f(m,h(r)) + k$	$h(r) \equiv h(\alpha^{g(m,s)} y_A^{f(m,h(r))})$
EG III.5	$f(m,h(r))$	$g(m,s)$	$f(m,h(r)) \equiv -x_A g(m,s) + k$	$h(r) \equiv h(\alpha^{f(m,h(r))} y_A^{g(m,s)})$
EG IV.1	$f(m,s)$	$g(h(r),s)$	$f(m,s) \equiv -x_A g(h(r),s) + k$	$h(r) \equiv h(\alpha^{f(m,s)} y_A^{g(h(r),s)})$
EG IV.6	$g(h(r),s)$	$f(m,s)$	$g(h(r),s) \equiv -x_A f(m,s) + k$	$h(r) \equiv h(\alpha^{g(h(r),s)} y_A^{f(m,s)})$
EG V.2	$f(m,h(r))$	$g(h(r),s)$	$f(m,h(r)) \equiv -x_A g(h(r),s) + k$	$h(r) \equiv h(\alpha^{f(m,h(r))} y_A^{g(h(r),s)})$
EG V.4	$g(h(r),s)$	$-f(m,h(r))$	$g(h(r),s) \equiv -x_A f(m,h(r)) + k$	$h(r) \equiv h(\alpha^{g(h(r),s)} y_A^{f(m,h(r))})$

Table 5: Efficient DSA-variants for signature verification

[HoP394] P.Horster, H.Petersen, "Classification of blind signature schemes and examples of hidden and weak blind signatures", Presented at the Rump Session of Eurocrypt '94, Perugia, Italy, (1994), 6 pages.[1]

[HoPM94] P.Horster, H.Petersen, M.Michels, "Meta signature schemes giving message recovery based on the discrete logarithm problem and some applications", (in preparation), 11 pages.[1]

[Knob93] H.-J.Knobloch, "A remark on the size of ElGamal-type digital signatures", Draft version, (1993), 5 pages.

[Knob94] H.-J.Knobloch, personal communication at the Eurocrypt'94 in Perugia, Italy, (1994).

[NIST91] National Institute of Standards and Technology, Federal Information Process. Standard, FIPS Pub XX: Digital Signature Standard (DSS), (1991).

[Nybe94] K.Nyberg, "Comment: New digital signature scheme based on discrete logarithm"', Electronics Letters, Vol. 30, No. 6, (1994), pp. 481.

[NyR193] K.Nyberg, R.Rueppel, "A new signature scheme based on the DSA giving message recovery", 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, Nov. 3–5., (1993), 4 pages.

[NyR293] K.Nyberg, R.Rueppel, "New signature schemes based on the discrete logarithm problem or how to add message recovery to the DSA", Preprint, Nov. 12, (1993), 13 pages.

[NyRu94] K.Nyberg, R.Rueppel, "Message recovery for signature schemes based on the discrete logarithm problem", Pre-proceedings of Eurocrypt '94, (1994), pp. 175 –190.

[Schn89] C.P.Schnorr, "Efficient identification and signatures for smart cards", Lecture Notes in Computer Science 435, Advances in Cryptology: Proc. Crypto '89, Berlin: Springer Verlag, (1990), pp. 339–351.

[YeLa93] S.-M. Yen, C.-S.Laih, "New Digital Signature Scheme based on Discrete Logarithm", Electronics Letters, Vol. 29, No. 12, (1993), pp. 1120–1121.

[1] Preprint available via anonymous FTP in directory *ftp.tu-chemnitz.de/pub/Local/informatik/crypto*.