Doing More with Fewer Bits

A.E Brouwer[1], R. Pellikaan[1] E. R. Verheul[2]

[1] Department of Math. and Comp. Sc., P.O. Box 513, Eindhoven University of Technology, 5600 MB, Eindhoven, The Netherlands. [aeb,ruudp]@win.tue.nl
[2] PricewaterhouseCoopers GRMS Crypto Group P.O. Box 85096, 3508 AB Utrecht Eric.Verheul@[nl.pwcglobal.com, pobox.com]

Abstract. We present a variant of the Diffie-Hellman scheme in which the number of bits exchanged is one third of what is used in the classical Diffie-Hellman scheme, while the offered security against attacks known today is the same. We also give applications for this variant and conjecture a extension of this variant further reducing the size of sent information.

1 Introduction

In the classical Diffie-Hellman key-exchange scheme, two system parameters are fixed: a large prime number P and a generator g of the multiplicative group of the basic finite field $GF(P)$. If two parties, Alice and Bob say, want to agree on a common secret key over an insecure channel, then Alice generates a random key $0 \leq x < P-1$ and sends $A = g^x \bmod P$ to Bob. Also, Bob generates a random key $0 \leq y < P-1$ and sends $B = g^y \bmod P$ to Alice. Both Alice and Bob can now determine the common, secret key $S = g^{xy} \bmod P = A^y \bmod P = B^x \bmod P$. For adequate security, P should be a 1024 bit prime, such that $P-1$ contains a 160 bit prime factor (see below). In particular this means that all system parameters (i.e. g, P) and sent information (i.e. A, B) are of size 1024 bits.

In [14], Claus Schnorr proposed a variant of the classical Diffie-Hellman scheme, in which g does not generate the whole multiplicative group of the basic finite field $GF(P)$, but only a small subgroup of which the order contains a 160 bit prime number q. As is suggested by Arjen Lenstra in [7], one can extend the Schnorr scheme to any multiplicative subgroup $G = \langle g \rangle$ of an extension field $GF(p^t)$. Lenstra specializes to generators g that have prime order, which we do as well.

For solving the discrete logarithm problem for a generator g of prime order q, one can use an index calculus (IC) based algorithm that has a heuristic expected asymptotic running time of $L(p^s, 1/3, 1.923 + o(1))$, see [1] and [7], where s is the smallest divisor of t such that $\langle g \rangle$ is contained in a subfield of $GF(p^t)$ isomorphic to $GF(p^s)$. If $p = 2$ then the constant 1.923 can be replaced by 1.587, see [3]. Alternatively one can use Birthday Paradox (BP) based algorithms (e.g. Pollard's rho algorithm [13]) that have expected running times exponential in the size of the q. More precisely, breaking the Discrete Logarithm problem can be solved in expected $O(\sqrt(q))$ elementary operations in $GF(p^t)$.

(1)

This leads us to the conclusion from [7] that - *w.r.t. attacks known today* - if the minimal surrounding subfield of g and prime order q of g are of sufficient size, then the discrete logarithm problem $\langle g \rangle$ is intractable. The particular form of the field itself, is not relevant. In other words, if $GF(p^t)$ is the minimal surrounding field of a subgroup of prime order, then - *w.r.t. attack known today* - the discrete logarithm in this subgroup is approximately as difficult as the discrete logarithm in a subgroup of prime order of a basic field $GF(P)$ if the size of P is approximately equal to as t times the size of p, and the order of both subgroups are about the same size. Hence, a suitable generator g in a field extension $GF(p^t)$ should not be contained in one of the proper subfields and should have a suitably large prime order. In practice, the size of the (minimal) surrounding field should be a ≥ 1024 bit number, and the prime order of the element should at least be of size ≥ 160 bits.
In [7], Lenstra proposes a simple, practical method for the construction of a field extension and suitable generator. The idea is that one fixes the size of the prime number p and a number t such that p^t is "large" enough. Then one looks for a large prime factor q in the value of the cyclotomic polynomial $\phi_t(p)$. The latter can be done using trial divisions with the primes up to, say 10^5; any other reasonable bound or method will do. Finally, one constructs a generator g of order q, by looking for an element different from 1, such that $g^{(p^t-1)/q} = 1$.

Using the above construction, the size of q is about $\varphi(t) \cdot |p|$ bits (where $\varphi(.)$ is Euler's totient function) which grows as least as fast as $p^{t/\log(log(t))}$. The complexity of the BP based algorithms grows much faster, than the complexity of the IC based algorithms. So if the size of the surrounding field is large enough to resist the sub-exponential, IC based algorithms, then the order ϖ will usually be large enough "automatically" to resist the BP based algorithms as well.

In this paper we propose a variant of the Diffie-Hellman scheme, using a multiplicative group G of an extension field $GF(p^6)$ as indicated above. For adequate security, the size of $GF(p^6)$ is a ≥ 1024 bit number and the order of $G = \langle g \rangle$ is a ≥ 160 bit prime factor of $\phi_6(p)$. Our scheme has the following properties:

1. Breaking our scheme, means breaking the Diffie-Hellman scheme in G, which in (today's) practice is as least as difficult as breaking the classical Diffie-Hellman scheme w.r.t. a modulus of comparable size.
2. All sent information (i.e. the A, B mentioned above) is only one third of the normal size, i.e. 342 bits. This makes our variant of the Diffie-Hellman scheme more competitive with Elliptic curve cryptosystems.

Outline
In Section 2, mainly as an appetizer, we will present a description of the variant of the Diffie-Hellman scheme based on Lucas sequences. In this variant all sent information (i.e. the A, B above) is only one half of the usual size (i.e. 512 bits in practice). In Section 3 we present an improvement of this scheme, in which all sent information is only one third of the usual size. A discussion of some applications of our scheme appears in Section 4, and in Section 5 we discuss and conjecture extensions of our scheme. We summarize our results in Section 6.

2 A different view of the LUCDIF cryptosystem

Central in the construction of the LUCDIF variant of the Diffie-Hellman key-exchange scheme is a $\geq$ 512-bit prime number p, such that $p+1$ contains a prime factor q of at least 160 bits. We now consider the field extension $GF(p^2)$, in which the multiplicative group is of order $p^2 - 1 = (p+1)(p-1)$. So we can construct an element g of order q in $GF(p^2)$; the prime numbers p, q and the generator g are the parameters of the LUCDIF system.
Now, if Alice and Bob want to agree on a common secret key, then Alice generates a random key $0 \leq x < q$, forms $A = g^x + g^{-x}$ (in the field $GF(p^2)$) and sends this to Bob. Similarly, Bob generates a random key $0 \leq y < q$, forms $B = g^y + g^{-y}$ and sends this to Alice. We will now first show that both Alice and Bob can determine the common, secret key $S = g^{xy} + g^{-xy}$.

For Alice to determine this shared secret key, she will first retrieve g^y from B. If we denote g^y by U, then we have the following equation

$$B = U + U^{-1} \tag{2}$$

Equality (2) is a quadratic equation in U (namely $U^2 - B * U + 1 = 0$) with coefficients in $GF(p)$ (see below). By using standard techniques (adjoining roots using the "abc-formula" in a symbolic way), Alice can find the two solutions $U_{1,2}$ in $GF(p^2)$ of this equation. All calculations in $GF(p^2)$ are symbolic and are effectively performed using operations in $GF(p)$.
These two solutions correspond exactly with g^y, g^{-y}. However, Alice has no idea which is g^y and which is g^{-y}, but that does not matter as she can determine $U_1^x + U_2^x$ by using her random key x. This is equal to $S = g^{xy} + g^{-xy}$, independent of the choice of U_1 and U_2. In a similar way, Bob can construct S from A. It is indicated in [2], that the above scheme coincides with the variant of the Diffie-Hellman key-exchange scheme that was proposed and analyzed by a series of authors; [15] (where the name 'LUCDIF' was proposed), [11], [10], [12] and [8].
We will now proceed with showing two important properties of the LUCDIF cryptosystem: reduced size of sent information and security.

2.1 Reduced size of sent information of the LUCDIF cryptosystem

For this property, we will show that both A and B are elements of the basic field $GF(p)$ and can therefore be represented by $|p|$ (e.g. 512) bits each. To this end, we first observe that any element h in the group generated by g, has the following property which we shall use often: $h^p = h^{-1}$. This follows as the order, q, of g (and therefore of h) divides $p+1$. We now have

$$A^p = (g^x + g^{-x})^p = g^{xp} + g^{-xp} = g^{-x} + g^x = A,$$

which means that A is an element of $\mathrm{G}F(p)$ as we needed to show. It similarly follows that B is an element of $GF(p)$ also.

It easily follows from the above discussion, that it suffices to take p, q and $g+g^{-1}$ as the system parameters instead of p, q and g. Hence, an additional advantage of the LUCDIF variant is that the size of the system parameters are also about halve the normal size. The same is realized in the setting of RSA in [6].

2.2 Security of the LUCDIF cryptosystem

Concerning the security, we will show that if somebody (an oracle $\mathcal{O}$) can break the LUCDIF system (i.e. determining S on basis of A and B) then one can break the Diffie-Hellman problem in the group $\langle g \rangle$ generated by g.
This would conclude the discussion security of the LUCDIF system. Indeed, the order q of g is a divisor of $p+1 = \phi_2(p)$ and we recall from the introduction that breaking the Diffie-Hellman scheme in $\langle g \rangle$ is - w.r.t. to attacks known today - is as infeasible as breaking the standard Diffie-Hellman scheme of a comparable size.

To this end, suppose $\alpha = g^x$, $\beta = g^y$ are given, then one can first construct $\alpha + \alpha^p$, $\beta + \beta^p$ and use this as input for $\mathcal{O}$ to determine $S_1 = g^{xy} + g^{xyp}$. By applying the same technique to α and $\beta \cdot g = g^{y+1}$ one can also determine $S_2 = g^{x(y+1)} + g^{x(y+1)p} = \alpha \cdot g^{xy} + \alpha^p \cdot g^{xyp}$. This means we have the following equation:

$$\begin{pmatrix} 1 & 1 \\ \alpha & \alpha^p \end{pmatrix} \cdot \begin{pmatrix} g^{xy} \\ g^{xyp} \end{pmatrix} = \begin{pmatrix} S_1 \\ S_2 \end{pmatrix} \qquad (3)$$

By this equation one now can deduce g^{xy}, i.e. the Diffie-Hellman shared secret key. Observe that the matrix above is regular because $\alpha \neq \alpha^p$ as α is not a member of $GF(p)$ by construction. Conversely, if somebody can break the Diffie-Hellman scheme in $\langle g \rangle$, then it is simple to show that one can break the LUCDIF system.

3 Our System

What is done in the LUCDIF scheme from an algebraic point of view, is representing an element z of the extension field $GF(p^2)$ not as the usual residue class modulo a fixed irreducible polynomial of degree 2, but by its unique, minimal polynomial, see [9]. If the element z is chosen in $\langle g \rangle$, then we can save information as the constant term of the minimal polynomial is always one, leaving only the first order coefficient (an element of $GF(p)$) to be stored or sent. However, the minimal polynomial of z does not only represent z, but also its conjugate $z^p = z^{-1}$. That is why we take the sum of the two conjugates (a symmetric function) to represent the exchanged key in the Diffie-Hellman scheme. We have shown that this is no problem with respect to security.

In this section we will develop a generalization of this technique in $GF(p^6)$. Central in our generalization is a 171-bit prime number p, such that the sixth cyclotomic polynomial $\phi_6(p) = p^2 - p + 1$ (see [9]) contains a prime factor q of at least 160 bits. As $\phi_6(p)$ is a divisor of $p^6 - 1$ (the order of the multiplicative group of $GF(p^6)$) we can easily construct a generator g in $GF(p^6)^*$ of order q. The prime numbers p, q, and the generator g are the parameters of our system. Actually, by taking a different representation of g the size of the system parameters can be reduced, see subsection 3.1.

Now, if Alice and Bob want to agree on a common secret key, then Alice generates a random key $0 \leq x < q$, and forms the minimal polynomial $P(X)$ of g^x, i.e.

$$P_A(X) = \prod_{i=0}^{5}(X - g^{xp^i}) = X^6 + A_5X^5 + A_4X^4 + A_3X^3 + A_2X^2 + A_1X + 1,$$

where all A_i are in $GF(p)$. Note that the constant term of $P_A(X)$ is 1 as $\phi_6(p)$ divides $1 + p + ... + p^5 = (p^6 - 1)/(p - 1)$. Then, Alice sends (A_1, A_2) to Bob. Bob also generates a random key $0 \leq y < q$, and forms the minimal polynomial $P(X)$ of g^y, i.e.

$$P_B(X) = \prod_{i=0}^{5}(X - g^{yp^i}) = X^6 + B_5X^5 + B_4X^4 + B_3X^3 + B_2X^2 + B_1X + 1 \quad (4)$$

where all B_i are in $GF(p)$. Bob then sends (B_1, B_2) to Alice.
The shared secret key will be the pair (C_1, C_2), i.e. the first and second order coefficients of the polynomial $P_C(X)$ given by:

$$P_C(X) = \prod_{i=0}^{5}(X - g^{xyp^i}) = X^6 + C_5X^5 + C_4X^4 + C_3X^3 + C_2X^2 + C_1X + 1$$

For Alice to determine this shared secret key, she will first retrieve the polynomial $P_B(X)$ from (B_1, B_2). To do this, she needs to reconstruct the B_3, B_4, B_5 from B_1, B_2. To this end, as $p^3 = -1 \bmod p^2 - p + 1$ it follows that the polynomial $P_B(X)$ is symmetric, i.e. $B_5 = B_1$ and $B_4 = B_2$.
It also follows that if we denote g^y by β, and $\beta_i = \beta^{p^i}$ for $i = 0, 1, 2, 3, 4, 5$ then

$$\beta_0 = \beta,\ \beta_1 = \beta^p,\ \beta_2 = \beta^{p-1},\ \beta_3 = \beta^{-1},\ \beta_4 = \beta^{-p},\ \beta_5 = \beta^{1-p}. \quad (5)$$

By equation (4) one can write B_3 in terms of the $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5$, and by using the reductions from (5), one can easily verify that:

$$B_3 = -2 \cdot \sum_{i=0}^{5} \beta_i - \sum_{i=0}^{5} \beta_i^2 - 2,$$

which is a symmetric polynomial in $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5$ of degree 2, and which can hence be written in the first and the second elementary polynomials of $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5$ by the so-called Newton equalities. Of course the value of the first symmetric polynomial equals $-B_1$ and the value of the second symmetric polynomial equals B_2. This leads to the following, easily verified formula.

$$B_3 = -2 + 2 * B_1 - B_1^2 + 2 * B_2.$$

So, starting from (B_1, B_2), Alice is able to retrieve the polynomial $P_B(X)$. She can then adjoin a root ρ of this polynomial to obtain $GF(p^6)$, next she can use her secret key x, to determine the minimal polynomial of ρ^x, which is equal to $P_C(X)$. That is, Alice is able to determine the shared secret key (C_1, C_2). Similarly, Bob is able to determine (C_1, C_2) from (A_1, A_2) and his secret key y.

3.1 Reduced Size Property of our System

Alice and Bob only send two coefficients in $GF(p)$ to each other, which corresponds to only $2|p|$ bits of data.

It easily follows from the above discussion, that it suffices to take p, q and the first and second order coefficients (elements of $GF(p)$) of the minimal polynomial of g, as system parameters. Hence, a typical size of the system parameters of our scheme would be 673 bits, consisting of $171+160 = 331$ bits for representing p and q plus 342 bits for representing g. The system parameters of a comparative Diffie-Hellman scheme would be 2048 bits (even 2208 bits for the Schnorr variant), which is more than three times as large.

3.2 Security of our system

We'll first show that the security of our variant of the Diffie-Hellman scheme is equivalent to the security of the Diffie-Hellman scheme in $\langle g \rangle$. This would conclude the discussion security of our system, as we recall that from the introduction that breaking the Diffie-Hellman scheme in $\langle g \rangle$ is - w.r.t. to attacks known today - is as infeasible as breaking the standard Diffie-Hellman scheme of a comparable size.
To this end, consider the following two functions $Z_1(.), Z_2(.) : \langle g \rangle \to GF(p^6)$ defined by:

$$Z_1(h) = \sum_{i=0}^{5} h^{p^i},$$
$$Z_2(h) = \sum_{0 \leq i \neq j \leq 5} h^{p^i + p^j}$$

Then, in the terminology of the previous section,

$$A_1 = Z_1(g^x), B_1 = Z_1(g^y), C_1 = Z_1(g^{xy})$$
$$A_2 = Z_2(g^x), B_2 = Z_2(g^y), C_2 = Z_2(g^{xy})$$

We have the following result, the proof of which is similar to the argument used in proving the security of the LUCDIF system in Section 2.2

Lemma 3.1 *For $i = 1, 2$, the problem of determining $Z_i(g^{xy})$ from $Z_i(g^x), Z_i(g^y)$ is as least as difficult as solving the Diffie-Hellman problem in $\langle g \rangle$.*

The security of our variant of the Diffie-Hellman scheme is equivalent to the difficulty of determining $Z_1(g^{xy}), Z_2(g^{xy})$ from,

$$Z_1(g^x), Z_1(g^y), Z_2(g^x), Z_2(g^y).$$

Hence, it immediately follows from Lemma 3.1 that solving this problem is equivalent with solving the Diffie-Hellman problem in $\langle g \rangle$. With respect to the attacks known today, this is just as difficult as breaking the Diffie-Hellman scheme in a basic field $GF(P)$ of size $|P| = 6 * |p|$.
So as we only need to exchange two coefficients of $|p|$ bits each, all sent information is one third of the size of the standard Diffie-Hellman scheme, while the offered security is the same.

It is shown in [16] that Lemma 3.1 is a consequence of a much broader result. To this end, let $t > 1$ be an integer ($t = 6$ in the current setting). Let n be a non-negative number and consider the integers $e_1, , e_n$ (the "exponents") and the elements $\lambda_1, ..., \lambda_n \in GF(p^t) \setminus \{0\}$ (the "multipliers") and consider the following *summing* function $Z(.) : \langle g \rangle \to GF(p^t)$ defined by:

$$Z(\kappa) = \sum_{i=1}^{n} \lambda_i \cdot \kappa^{e_i}, \text{for } \kappa \in \langle g \rangle$$

The number n is called the *degree* of the summing function and the number $d = \gcd(e_1, e_2, ..., e_n, \mathrm{ord}(g))$ is called the *order* of the summing function. It is a simple verification, that the above introduced $Z_1(.), Z_2(.)$ are summing functions of order 1.
Now the following "hardness" result is shown in [16].

Theorem 3.2 *In the above terminology, let $Z(.)$ be a summing function of order d. Also let $\mathcal{O}$ be an oracle that on basis of any γ^x and γ^y computes $Z(\gamma^{xy})$. Then there exists a polynomial time algorithm that computes γ^{xyd} on basis of γ^x and γ^y. That is, for $d = 1$ there exists a polynomial time algorithm that solves the whole Diffie-Hellman problem in $\langle \gamma \rangle$.*

3.3 Implementation of our system

In determining a shared secret key in our system a participant typically has to perform the following operations:

1. Restore the minimal polynomial he received, and adjoin a root ρ to $GF(p)$ satisfying this polynomial, giving a copy of the field $GF(p^6)$.

2. Determining ϕ, i.e. ρ raised to his random key in the representation given by the root ρ and its minimal polynomial, i.e. as a linear combination of ρ^i, with $i = 0, 1, 2, 3, 4, 5$.
3. Determine the values of the first and second order coefficients of the minimal polynomial of ϕ.

The first operation is of negligible complexity. For the second operation a repeated square-and-multiply algorithm should be used, taking $O(36 \cdot \ln(q) \cdot \ln(p)^2)$ bit operations. For the final step, the representations of the conjugates of ϕ, i.e. ϕ^{p^i} for $i = 1, 2, 3, 4, 5$, have to be determined. As $p^3 = -1 \bmod p^2 - p + 1$, only ϕ^p, ϕ^{p^2} need to be calculated. As furthermore

$$\phi^{p^2-p+1} = \phi^{p^2} \cdot \phi^{-p} \cdot \phi = 1$$

it follows that only the representation of ϕ^p needs to be calculated with a repeated square-and-multiply algorithm; the representations of the remaining conjugates can be determined by taking inverses. This takes an additional $O(36 \cdot \ln(q) \cdot \ln(p)^2)$ bit operations. The values of the first and second order coefficients of the minimal polynomial of ϕ can now be easily determined as the values of the first and second elementary symmetric polynomials of the conjugates of ρ. Given the representation of these conjugates, this is of negligible complexity. All and all, determining a shared secret key in our system by a participant takes $O(36 \cdot \ln(q) \cdot \ln(p)^2)$ bit operations, which is at least asymptotically comparable with the number of operations needed for a Diffie-Hellman key-exchange in a basic field of comparable size.

It's rather unfortunate that it seems that we can not fix the representation of $GF(p^6)$ to the one proposed by Arjen Lenstra in [7], in which an optimal normal basis can be used, such that exponentiation can carried out even more efficiently than in a basic field of the same size.

4 Applications of our system

Our system can not only be used to obtain a variant of the Diffie-Hellman scheme where the exchanged data is one third of the usual size, but it can also be used to construct variants of schemes which are related to this scheme such as the ElGamal encryption scheme [4]. In this variant Bob's public key takes the form $(Z_1(y), Z_2(y))$, where $y = g^x$ and where $0 \leq x < q$ is Bob's private key. An encryption for Bob (e.g. by Alice) of an element S in $GF(p)$ takes the form

$$[(Z_1(g^k), Z_2(g^k)), S + Z_1(y^k)]$$

where k is taken randomly less than q. It follows from the discussion of the previous section, that Alice is indeed capable to determine this encryption, and Bob is indeed capable to decrypt this and recover S. It easily follows that the ability to break this system, means the ability to solve the Diffie-Hellman system

in $\langle g \rangle$, which is, as far as is known today, just as difficult as breaking the Diffie-Hellman scheme in a basic field $GF(P)$ of size $|P| = 6 * |p|$.
Of course, many other variants are possible, such as using $Z_2(.)$ instead of $Z_1(.)$. In this variant the size of the public keys (342 bits) is one third of the "normal" size. Moreover, the total encryption of an element of $GF(p)$ takes a total size of $3 * 171 = 513$ bits. So, when using hybrid encryption (only encrypting a random session key asymmetrically) for a non-interactive application (e.g. email), the data requirement for our ElGamal variant is even less than for an RSA encryption of the same security. Actually, as in such cases only a random session key is required, one can use $Z_1(y^k)$ (with k random) as the session key, and include $(Z_1(g^k), Z_2(g^k))$, with the message. In this fashion, only one third of the size is required for the asymmetric part of what is usually used for an RSA encryption.

Apart from asymmetric schemes for confidentiality, the scheme can be used to make variants of digital signature schemes like the Digital Signature Algorithm [5], with public keys that are only a third of the usual size. The idea is that a verification of type $v = g^{u_1} \cdot y^{u_2}$, occurring in the verification part of the Digital Signature Algorithm where v, u_1, u_2 are constructed from the digital signature, can be recovered from $(Z_1(g), Z_2(g))$ and $(Z_1(y), Z_2(y))$, albeit not in a unique fashion: there are $6 \cdot 6 = 36$ possibilities, leading to 36 verifications one of which should hold. Of course, several straight-forward techniques can be employed to reduce this number of verifications. For instance, at the cost of a one time distribution of the (whole) generator g, instead of its minimal polynomial, one can reduce the number of verifications to six.

5 Extensions of our system

Consider a prime number p and an integer t such that:

1. the multiplicative group of $GF(p^t)$ is large enough to withstand the Index Calculus based attacks;
2. the t-th cyclotomic polynomial $\phi_t(p)$ contains a prime factor q that is large enough to withstand the Birthday Paradox based algorithms.

Next a generator g is chosen of order q. Any element $h \in \langle g \rangle$ can be represented as an element of $GF(p^t)$ using $t \cdot |p|$ bits. However, the degree of the t-th cyclotomic polynomial is equal to $\varphi(t)$, where $\varphi(.)$ is Euler's totient function. So, in principle only $\varphi(t) \cdot |p|$ bits are required to represent any element $h \in \langle g \rangle$. A straightforward way to do this is to write $h = g^x$ for some $0 \leq x < q$ and to represent h by x. However, this means solving the discrete logarithm problem for h with respect to g. What we aim for, is a technique to represent ("compress") h by only $\varphi(t) \cdot |p|$ bits, without solving the discrete logarithm problem for h with respect to g. With this technique, Alice and Bob can agree on a common secret key in $\langle g \rangle$ by sending each other only $\varphi(n) \cdot |p|$ bits.

Motivated by the techniques from Sections 2 and 3 we conjecture the following "compressed" form of the Diffie-Hellman scheme.

If Alice and Bob want to agree on a common secret key, then Alice generates a random key $0 \leq x < q$, and forms the minimal polynomial $P_A(X)$ of g^x and somehow represents this with using only $\varphi(t) \cdot |p|$ bits and sends this to Bob. Similarly, Bob generates a random key $0 \leq y < q$, and forms the minimal polynomial $P_B(X)$ of g^y and somehow represents this with using only $\varphi(t) \cdot |p|$ bits and sends this to Alice. Using this representation, Alice is able to determine the minimal polynomial $P_B(X)$, and to adjoin a root ρ of $P_B(X)$ to $GF(p)$ and to determine a representation of $GF(p^t)$. Using this she is able to determine the minimal polynomial $P_C(X)$ in $GF(p)$ of ρ^x, which is equal to the minimal polynomial of g^{xy}. Alice uses the coefficient S of the first order term in $P_C(X)$ (i.e. the sum all conjugates of g^{xy}) as a secret key. Similarly, Bob is able to determine S using the representation of $P_A(X)$ and his private key.

With such a scheme, one obtains a version of the Diffie-Hellman scheme in which one only sends a $\frac{\varphi(t)}{t}$ part of the information one would normally send. If t is the product of the first k prime numbers, then one can easily show that this fraction goes to zero if k goes to infinity. So the achieved reduction gets better all the time. Also, using Theorem 3.2 one can show that (given such a representation) breaking this scheme, means solving the Diffie-Hellman problem in $\langle g \rangle$, which is suitable intractable, as far as is known today. Moreover, several other, secure choices (values of a symmetric function in all conjugates of g^{xy}) for the shared secret key S are possible.

Of course, the problem is how to represent such minimal polynomials by only $\varphi(t) \cdot |p|$ bits. In Sections 2 and 3 we have given such representations for $t = 2$ and $t = 6$. If t is a prime number r, then such a representation is straightforward. Indeed, as $\phi_t(p)$ divides $(p^t - 1)/(p - 1)$, the constant term of the minimal polynomials is 1. Hence only $r - 1 = \varphi(r)$ coefficients are unknown, each of size $|p|$ bits.

Moreover, the representation used for $t = 6$ can be extended to the case where t is of type $2r$ where r is prime as follows. Let, as before, $P_A(X)$ be the minimal polynomial of degree t of g^x in $GF(p)$. Then, $P_A(X)$ splits as a product of two polynomials $P_{A_1}(X)$, $P_{A_2}(X)$ of degree r in $GF(p^2)$. Let us denote:

$$P_{A_1}(X) = X^r + a_{r-1,1}X^{r-1} + ...a_{1,1}X + 1 \tag{6}$$

$$P_{A_2}(X) = X^r + a_{r-1,2}X^{r-1} + ...a_{1,2}X + 1. \tag{7}$$

The constant terms of both polynomials are 1 as $\phi_{2r}(p)$ divides $(p^{2r}-1)/(p^2-1)$. Then it easily follows that

$$(a_{i,1})^p = a_{i,2} \text{ for all } i = r - 1, ..., 2. \tag{8}$$

It also follows that the reciprocal polynomial of $P_{A_1}(X)$ coincides with $P_{A_2}(X)$, i.e.:

$$X^r \cdot P_{A_1}(1/X) = P_{A_2}(X). \tag{9}$$

Now, suppose one possesses the first $(r-1)/2$ non-trivial coefficients of $P_{A_1}(X)$, i.e. $a_{r-1,1}, ..., a_{(r-1)/2-1,1}$, then using formula (8) one obtains the first $(r-1)/2$

non-trivial coefficients of $P_{A_2}(X)$, i.e. $a_{(r-1)/2,2}, ..., a_{(r-1)/2-1,2}$. From these and formula (9) it follows that one obtains the remaining $(r-1)/2$ non-trivial coefficients of $P_{A_1}(X)$, i.e. $a_{(r-1)/2,1}, ..., a_{1,1}$.
That is, from the first $(r-1)/2$ non-trivial coefficients of $P_{A_1}(X)$ one can reconstruct $P_{A_1}(X)$, and hence $P_A(X)$.

Hence, we can represent $P_A(X)$ by $(r-1)/2$ coefficients in $GF(p^2)$ for which one only needs $(r-1)/2 \cdot 2|p| = (r-1)|p| =$ bits, which is in accordance with our conjecture.

Problem How to find representations of minimal polynomials of only $\varphi(t) \cdot |p|$ bits for general t?

Irrespective of their existence, we note that the number of extensions of our system, as discussed above, that are more efficient in practice is quite low. To illustrate, assuming that for the coming years a classical (e.g. RSA) asymmetric key length between 1024 and 2048 bits gives adequate security for most (commercial) applications, then there are actually only two possible more efficient practical extensions. The first one, corresponds with t equal to $30 = 2 \cdot 3 \cdot 5$, where a reduction of $\varphi(30)/30 = 4/15$ can then be achieved. The characteristic of the used field would be a prime number between 35 and 70 bits length. The second one, corresponds with t equal to $210 = 2 \cdot 3 \cdot 5 \cdot 7$, where a reduction of $\varphi(210)/210 = 8/35$ can then be achieved. The characteristic of the used field would be a prime number between 5 and 10 bits length. For t equal to $2310 = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11$, the used field size would have to larger than a 2048 bit number.

6 Conclusion

We have presented a variant of the Diffie-Hellman scheme in which all sent information is one third of the size of the standard Diffie-Hellman scheme, while the offered security, as far as is known today, is the same. We have also given applications for this construction. Finally, we have given a conjecture for an extension of our scheme in which all sent information is only a factor $\varphi(t)/t$ of the size of the standard Diffie-Hellman scheme.

References

1. M. Adleman, J. DeMarrais, *A subexponentional algorithm over all finite fields*, CRYPTO '93 Proceedings, Springer-Verlag, pp. 147-158.
2. D. Bleichenbacher, W. Bosma, A.K. Lenstra, *Some remarks on Lucas-Based Cryptosystems*, CRYPTO '95 Proceedings, Springer-Verlag, pp. 386-396.
3. D. Coppersmith, *Fast evaluation of logarithms in fields of characteristic two*, IEEE Transactions on Information Theory, 30, (1984), pp. 587-594.
4. T. ElGamal, *A Public Key Cryptosystem and a Signature scheme Based on Discrete Logarithms*, IEEE Transactions on Information Theory 31(4), 1985, pp. 469-472.

5. FIPS 186, *Digital signature standard*, Federal Information Processing Standards Publication 186, U.S. Department of Commerce/ NIST, 1994.
6. A.K. Lenstra, *Generating RSA moduli with a predetermined portion*, Asiacrypt '98 proceedings, Springer-Verlag, pp. 1-10.
7. A.K. Lenstra, *Using Cyclotomic Polynomials to Construct Efficient Discrete Logarithm Cryptosystems over Finite Fields*, Information Security and Privacy - ACISP97 Proceedings (Sydney 1997), Lect. Notes in Comp. Sci. 1270, Springer-Verlag, pp. 127-138.
8. R. Lidl, W.B. Müller, *Permutation Polynomials in RSA-cryptosystems*, Crypto '83 Proceedings, Plemium Press, pp. 293-301.
9. R. Lidl, H. Niederreiter, *Finite Fields*, Addison-Wesley, 1983.
10. W.B. Müller, *Polynomial functions in modern cryptology*, Contributions to general Algebra 3, Proceedings of the Vienna Conference (1985), pp. 7-32. Proceedings, Springer-Verlag, pp. 50-61.
11. W.B. Müller, W. Nöbauer, *Cryptanalysis of the Dickson-Scheme*, Eurocrypt '85 Proceedings, Springer-Verlag, pp. 50-61.
12. W. Nöbauer, *Cryptanalysis of the Rédei Scheme*, Contributions to general Algebra 3, Proceedings of the Vienna Conference (1985), pp. 255-264.
13. J.M. Pollard, *Monte Carlo methods for index computation (*$\mathrm{mod}\, p$*)*, Mathematics of Computation, 32, (1978), pp. 918-924.
14. C.P. Schnorr, *Efficient signature generation by smart cards*, Journal of Cryptology, 4, pp. 161-174 (1991).
15. P. Smith, C. Skinner, *A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms*, Asiacrypt '94 proceedings, Springer-Verlag, pp. 357-364.
16. E.R. Verheul, *Certificates of Recoverability with Scalable Recovery Agent Security*, in preparation.