Int. J. Appl. Math. Comput. Sci., 2004, Vol. 14, No. 2, 265–267

A NEW APPROACH TO THE ELGAMAL ENCRYPTION SCHEME

CZESŁAW KOŚCIELNY*

* Academy of Management of Legnica, Faculty of Computer Science
ul. Reymonta 21, 59–220 Legnica, Poland
e-mail: C.Koscielny@wsm.edu.pl

The ElGamal encryption scheme can be used for both digital signatures and encryption, and its security results from the difficulty of calculating discrete logarithms in a finite field. This algorithm usually works in a multiplicative group of $GF(p)$ and in this case the progress in the discrete logarithm problem forces the users of such a basic ElGamal public key cryptosystem to permanently increase a prime modulus p in order to ensure the desired security. But the task of finding a multiplicative group of $GF(p)$ is unfeasible for an ordinary user. It is possible to overcome this inconvenience by forming an ElGamal encryption scheme which works in a multiplicative group of $GF(p^m)$. Therefore, it is shown in the paper how to implement this cryptosystem for work in the multiplicative group of $GF(p^m)$, in its subgroup, and in an algebraic system named the spurious multiplicative group of $GF(p^m)$.

Keywords: public-key encryption, ElGamal cipher, block ciphers

1. Introduction

Public-key cryptographic algorithms are designed to resist chosen-plaintext attacks and their security is based both on the difficulty of finding the secret key from the public key and the difficulty of determining the plaintext from the cryptogram. At present, the most common public-key cryptosystem is the RSA algorithm. It is conjectured that the security of RSA depends on the problem of factoring large numbers. It has never been mathematically proven that you need to factor the modulus n to calculate a plaintext knowing a cryptogram and a public key $\{e,\ n\}$. It is conceivable that an entirely different way to break RSA can be discovered (perhaps this way is already known for some cryptanalysts). Therefore, cryptographers attempt to activate alternative public-key encryption algorithms, e.g. the basic ElGamal encryption scheme.

It is well known that the progress in the discrete logarithm problem forces the users of the basic ElGamal public key cryptosystem, working in a multiplicative group of $GF(p)$, to permanently increase a prime modulus p in order to ensure the desired security. For long-term security, at least 2000-bit moduli should be used at present. Common system-wide parameters need even larger key sizes, since computing the database of discrete logarithms for one particular p will discredit the secrecy of all private keys computed using p. But the task of finding a multiplicative group of $GF(p)$ is infeasible for an ordinary user if $p > 2^{2000} \approx 0.115 \times 10^{603}$. As shown in the sequel, it is possible to overcome this inconvenience by forming an ElGamal public key cryptosystem which works in a multiplicative group of $GF(p^m)$ provided that a primitive polynomial $p(x)$ of degree m over $GF(p)$ will be used to construct $GF(p^m)$. Since the root of the primitive polynomial $p(x)$ is known, one can easily determine any generator of a multiplicative group of $GF(p^m)$, viz. any primitive element of this field. As it turns out, the cipher works also in an arbitrary subgroup of $GF(p^m)$, and even in such an algebraic system which awkwardly simulates the multiplicative group of $GF(p^m)$. Thus, the latter algebraic system, named the spurious multiplicative group of $GF(p^m)$, briefly denoted by $SMG(p^m)$, is described.

2. Implementation of the ElGamal Encryption Scheme in a Multiplicative Group of $GF(p^m)$

Although it is known that the basic ElGamal encryption scheme can be generalized to work in any finite cyclic group, particularly in a multiplicative group of $GF(p^m)$, that the operations on elements of the multiplicative group of $GF(p^m)$, namely, multiplication and exponentiation, are easy to implement and that the discrete logarithm problem in this group should be computationally infeasible (Menezes *et al.*, 1998, Stinson, 1995), no serious prac-

tical approach to implement such a system can be found in the literature. Therefore, the ElGamal encryption scheme, based on the multiplicative group of $GF(p^m)$, is considered in this paper.

A concise description of slightly modified algorithms, characterizing the presented approach, is given below.

Key generation: Each entity creates its public key and the corresponding private key. So each entity $\mathcal{A}$ ought to do the following:

- Choose a primitive polynomial $p(x)$ of the degree m over $GF(p)$ in order to construct a group, isomorphic to the multiplicative group of $GF(p^m)$. The group having the order $n = p^m - 1$, consists of the set $G = \{1, \ \ldots, \ p^m - 1\}$ and of an operation of the multiplication of elements from this set, which is performed by means of a function $P(x, y)$, $x, y \in G$. The function $Pow(x, k)$, carrying out the operation of rising any element from G to a k-th power, $k \in [-n+1, n-1]$, is also defined. The generator of this group is p, since the polynomial $p(x)$ is primitive.
- Select a random integer $r \in [1, n-2]$, such that $(r, n-1) = 1$, and compute the other generator $\alpha = Pow(p, r)$.
- Choose a random integer $a \in G$, and compute $\beta = Pow(\alpha, a)$.
- $\mathcal{A}'$s public key is α and β, together with $p(x)$ and functions P and Pow, if these last three parameters are not common to all entities.
- $\mathcal{A}'$s private key is a.

Encryption: Entity $\mathcal{B}$ encrypts a message m for $\mathcal{A}$, which $\mathcal{A}$ decrypts. Thus $\mathcal{B}$ should make the following steps:

- Obtain $\mathcal{A}'$s authentic public key $\alpha, \ \beta$, and $p(x)$ together with functions P and Pow if these parameters are not common.
- Represent the message m as a number from the set G.
- Choose a random integer $k \in [1, n-1]$.
- Determine numbers $c_1 = Pow(\alpha, k)$ and $c_2 = P(m, Pow(\beta, k))$.
- Send the ciphertext $c = c_1, \ c_2$ to $\mathcal{A}$.

Decryption: To find plaintext m from the ciphertext $c = c_1, \ c_2$, $\mathcal{A}$ should perform the following operations:

- Use the private key a to compute $g = Pow(c_1, a)$ and then compute $g^{-1} = Pow(g, -1)$.
- Retrieve the plaintext by computing $m = P(g^{-1}, c_2)$.

3. Remarks on the Implementation of the ElGamal Encryption Scheme Using a Multiplicative Group of $GF(p^m)$, Its Subgroup and a Spurious Multiplicative Group of $GF(p^m)$

It is assumed that to construct a multiplicative group of $GF(p^m)$ we will use primitive polynomials, but to determine an arbitrary primitive polynomial of a higher degree is not that simple. There exist a few useful tables, cf. e.g. (Živković, 1994), where about 600 primitive polynomials of degree from 311 to 3604 over $GF(2)$ are listed. It is a pity that it is difficult to find similar tables for odd p. Nevertheless, using computer–algebra systems one may easily determine a random irreducible polynomial of degree m over $GF(p)$, with p odd, satisfying $p^m \in [2^{1000}, 2^{5000}]$, and work with the ElGamal encryption scheme based on an arbitrary subgroup of the multiplicative group of $GF(p^m)$ with the key size 1000 – 5000 bits long.

A detailed description of the implementation of the ElGamal encryption scheme in Maple 8 using a multiplicative group of $GF(p^m)$, its subgroup and a spurious multiplicative group of $GF(p^m)$ is given in (Kościelny, 2003). In all these three cases we can use the same routines P and Pow and the primitive polynomial, irreducible polynomial, and an arbitrary polynomial over $GF(p)$, respectively. Since the notion of a spurious multiplicative group of $GF(p^m)$ is rather new, a concise introduction to this algebraic system is presented in the next section.

4. $SMG(p^m)$ — a Spurious Multiplicative Group of $GF(p^m)$

For all prime p, for any positive integer m and for any polynomial $f(x)$ of degree m over $GF(p)$ there exists an agebraic system $\langle Gx, \ \cdot \rangle$, consisting of the set Gx of all $p^m - 1$ non-zero polynomials of degree $dg \leq m - 1$ over $GF(p)$ and of an operation of the multiplication of these polynomials modulo polynomial $f(x)$. Such an algebraic system is a generalization of the multiplicative group of $GF(p^m)$, since the elements of the set Gx are the same as the elements of the multiplicative group of $GF(p^m)$. Therefore, it will be called the spurious multiplicative group of $GF(p^m)$ and will be denoted by

$SMG(p^m)$. $SMG(p^m)$ is obtained using the mapping σ, defined by the function $\sigma(v(x)) = v(p)$, converting a polynomial $v(x)$ belonging to the set Gx into a number from the set G. It is then clear that multiplication in $SMG(p^m)$ can be performed by the same routines or by the same hardware as in the multiplicative group of $GF(p^m)$, but the interior of the multiplication table of $SMG(p^m)$ is not a Latin square. In principle,

Table 1. Table of operation in $SMG(2^3)$ constructed using $f(x) = x^3 + 1$ and the mapping σ.

$\cdot$	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7
2	2	4	6	1	3	5	7
3	3	6	5	5	6	3	0
4	4	1	5	2	6	3	7
5	5	3	6	6	3	5	0
6	6	5	3	3	5	6	0
7	7	7	0	7	0	0	7

$SMG(p^m)$ is a commutative groupoid, in which the operation may be neither closed, nor fully associative, since divisors of 0 exist if $f(x)$ is not irreducible. The existence of such groupoids, which seems to be important for cryptography, has been noticed by the author recently (maybe other cryptographers experimented with similar algebraic systems). Thus, all their properties are not yet known to him. However, using the ElGamal cipher based on $SMG(p^m)$, we must remember that there exist non-zero elements belonging to this groupoid which do not have their multiplicative inverses. The set of all reversible elements forms an Abelian group under multiplication (function P), which is not generally cyclic. It is then easy to observe that any message can be encrypted by means of the ElGamal cipher based on $SMG(p^m)$, but the sender, after the cryptogram $c = c_1, c_2$ has been generated, should verify if c_1 is reversible, i.e. if $Pow(c_1, -1)$ exists and can be computed. It is a necessary condition for succesful deciphering.

An example of the table of operation in $SMG(2^3)$, constructed using the polynomial $x^3 + 1 = (x + 1)(x^2 + x + 1)$ is given in Tab. 1. For example, let us multiply x by $x^2 \pmod{x^3 + 1}$. The result will be 1. But since in the case considered $p = 2$, we have $\sigma(x) = 2$ and $\sigma(x^2) = 4$, and therefore $2 \cdot 4 = 4 \cdot 2 = 1$. Similarly, $(x + 1) \cdot (x^2 + x + 1) = x^3 + 1 \equiv 0 \pmod{x^3 + 1}$, and thus $7 \cdot 3 = 3 \cdot 7 = 0$.

The reader can find more details concerning the construction of an arbitrary $SMG(p^m)$ in (Kościelny, 2003).

5. Conclusions

It has been shown that the ElGamal public-key encryption algorithm based on a multiplicative group of $GF(p^m)$, on its subgroups and on an $SMG(p^m)$ is user-friendly, because in such a scheme the key-generation and encryprion/decryption algorithms are simple and effective, even if a key size of order up to 10000 bits or more is needed. Particularly interesting seems to be the use of $SMG(p^m)$, which makes the cryptosystem partly unpredictable and expands its keyspace. In the latter case, if we use, e.g. $SMG(2^{20000})$, we can construct about $2^{20000} \approx 0.398 \times 10^{6021}$ spurious multiplicative groups in which the 20000 bit key size fast ElGamal cryptosystem can work. However, the problem presented here is quite new and far from being fully discussed.

References

Kościelny C. (2003): *User-friendly ElGamal public-key encryption scheme*. — `http://www.mapleapps.com/List.asp?CategoryID=6&Category=Cryptography`

Menezes A.J., van Oorschot P.C. and Vanstone S.A. (1998): *Handbook of Applied Cryptography*. — Boca Raton: CRC Press.

Stinson D.R. (1995): *Cryptography — Theory and Practice*. — Boca Raton: CRC Press.

Živković M. (1994): *Table of primitive binary polynomials, Part II*. — Math. Comput., Vol. 63, No. 207, pp. 301–306.

Received: 16 November 2003