On Using Mersenne Primes in Designing Cryptoschemes

Moldovyan Alexander Andreevich[1], Moldovyan Nicolay Andreevich[1], and Berezin Andrey Nickolaevich[2]

(Corresponding author: Moldovyan Nicolay Andreevich)

Department of Secure Information Technologies, ITMO University[1]
Kronverksky pr., 10, St. Petersburg 197101, Russian Federation
Department of Automated Infor. Processing & Control Systems, St. Petersburg State Electrotechnical University[2]
ul. Professora Popova, 5, St. Petersburg 197376, Russian Federation

(Received July 15, 2013; revised and accepted Jan. 10 & May 9, 2014)

Abstract

The paper proposes justification of using Mersenne primes in the following cryptoschemes: commutative and public-key encryption algorithms and zero-knowledge protocol. The cryptoschemes are based on computational difficulty of finding discrete logarithm in the finite fields $GF(2^s)$, where s is a sufficiently large prime such that 2^{s-1} is also a prime, for example $s = 1279$, $s = 2203$, and $s = 4253$.

Keywords: Binary polynomials, commutative encryption, discrete logarithm problem, finite fields, Mersenne primes, public-key encryption, zero-knowledge protocol

1 Introduction

Computational difficulty of the discrete logarithm problem (DLP) in the finite fields is used in different cryptographic schemes such as public key distribution protocols [2], digital signature protocols [14], blind and collective signature protocols [10, 11], public-encryption algorithms, commutative-encryption algorithms [13], zero-knowledge protocols for user authentication [7] *et al.* The finite fields $GF(2^s)$ present some essential advantages for implementing the cryptoschemes: i) computational complexity of the multiplication operation is comparatively low; ii) for arbitrary $n > 1$ the set of all possible n-bit data blocks can be interpreted as elements of the field $GF(2^n)$. The multiplication operation in $GF(2^s)$ is performed as multiplying binary polynomials modulo an irreducible binary polynomial of the degree s. This operation is especially fast in the case of using low weight irreducible binary polynomials, for example $x^s + x^k + 1$, where $k < s/2$.

In this paper it is proposed a new design for zero-knowledge protocols, concrete variants of such protocols, and a new implementation of the commutative and public-key encryption algorithm. The proposed cryptoschemes are characterized in using such finite fields $GF(2^s)$ in which the order of their multiplicative group is a Mersenne prime $2^s - 1$. It is justified that this is the best case for selecting the value s.

2 Zero-knowledge Protocol

Any public key agreement cryptoscheme can be transformed into some zero-knowledge protocol. The idea of such transformation relates to the possibility of two users' generating a common secrete value with the help of their public key exchange. For example, in the *Diffie-Hellman* scheme the public key y is generated as follows, using sufficiently large prime p, such that some another large prime q divides the number $p - 1$, and a primitive element α modulo p. Some user generates his private key as a random value $k < p - 1$ and computes the value

$$y = \alpha^k \bmod p.$$

Another user generates his private key as a random value $u < p - 1$ and computes his public key $R = \alpha^u \bmod p$. Each of them is able to compute the common secret

$$Z = y^u \bmod p = R^k \bmod p.$$

Any other person is not able to compute Z until the DLP modulo p is solved and value k or value u is computed from the known values p, α, y, and R. Suppose the first user (claimant) is the person to be authenticated by the second user (verifier). Suppose also the verifier has been provided with a trusted copy of claimant's public key y. The proposed zero-knowledge protocol, in which there is used some specified hash function $h(*)$, includes the following two steps:

1) The verifier generates a random number $u < p - 1$ and computes the one pad public key $R = \alpha^u \bmod p$. Then he computes common secret Z related to R and

claimant's public key y: $Z = y^u \bmod p$, and hash function value $H = h(Z)$. Then verifier sends to claimant the pair of numbers (R, H) that is verifier's request.

2) Using his private key k the claimant computes the values $Z' = R^k \bmod p$ and $H' = h(Z')$. After that he compares the values H' and H. If $H' = H$, then the claimant sends to verifier the value Z' that is claimant's response, otherwise he sends to verifier the message "The request (R, H) is not correct".

On receipt of the response Z' the verifier compares Z' with the value Z. If $Z' = Z$, then the verifier accepts the claimant as valid owner of the public key y, otherwise the procedure fails. Let us note that computing the hash function values and comparison of the values H' and H performed by the claimant at Step 2 is sufficiently important. The identity of the values H' and H proves that the verifier has computed correctly the value R and knows the value Z', i.e. no information about the private key x is provided to the verifier with the response Z'. Computation of the hash function values at the first and second steps prevents the following attack on the claimant's private key. The verifier selects a value R' having sufficiently small prime order ω modulo p and sends R' as his request to the claimant. After receiving the response $Z' = R'^k \bmod p$ the verifier will be able to compute with the baby-step-giant-step algorithm [9] the value $k' \equiv k \bmod \omega$. If the value $p-1$ contains many small prime divisors r_i, for example $i = 1, 2, ..., g$, then such attack can be performed for the case of sufficiently large composite value that leads to computing the value k. Regarding this attack the most secure prime value p is such that $p = 2q + 1$, where q is a prime. For such modulus p this attack provides to attacker only one bit of the private key.

Thus, using computations in the ground field $GF(p)$ one should additionally specify the stage of computing hash function values from Z and Z' in the proposed zero knowledge protocol. To avoid this computation stage it is possible to construct a variant of such protocol using computations over binary polynomials modulo an irreducible polynomial, i.e. to define the zero-knowledge protocol over the binary field $GF(2^s)$ in which the multiplicative group has prime order $2^s - 1$ that is one of sufficiently large Mersenne primes. There are known the following values s to which correspond appropriate Mersenne primes: $s = 1279$; 2203; 2281; 3217; 4253; 4423; 9689; 9941; 11213 [1]. Other Mersene primes corresponding to values $s \geq 19937$ represent less interest for the considered application ($s = 19937, 21701, 23209, 44497...$ [1]).

In the public key agreement scheme over the field $GF(2^s)$ the public key is computed as follows $y(x) = (\alpha(x))^k \bmod p(x)$, where $p(x)$ is some irreducible binary polynomial of the degree s; $y(x)$ and $\alpha(x)$ are elements of $GF(2^s) different from 0 and 1$; $k < 2^s - 1$. Respectively we come to the following protocol that is free from using the hash functions.

1) The verifier generates a random number $u < 2^s - 1$ and computes the one pad public key $R(x) = (\alpha(x))^u \bmod p(x)$. Then he computes common secret $Z(x)$ related to $R(x)$ and claimant's public key $y(x) : Z(x) = (y(x))^u \bmod p(x)$. Then verifier sends to claimant the binary polynomial $R(x)$ that is verifier's request.

2) Using his private key k the claimant computes the values $Z'(x) = (R(x))^k \bmod p(x)$ and sends to verifier the value Z' that is claimant's response.

To reduce the computational complexity of the protocol one can use the low weight irreducible polynomials $p(x)$. Table 1 shows the variants of such polynomials.

Table 1: Low weight irreducible binary polynomials [8, 15]

Mersenne exponent	$p(x)$
1279	$x^{1279} + x^{216} + 1; x^{1279} + x^{418} + 1$
2203	$x^{2203} + x^{14} + x^6 + x^5 + 1$
2281	$x^{2281} + x^{715} + 1$; $x^{2281} + x^{915} + 1$; $x^{2281} + x^{1029} + 1$
3217	$x^{3217} + x^{67} + 1$; $x^{3217} + x^{576} + 1$
4253	$x^{4253} + x^{21} + x^{12} + x^{11} + 1$
4423	$x^{4423} + x^{271} + 1$; $x^{4423} + x^{369} + 1$; $x^{4423} + x^{370} + 1$; $x^{4423} + x^{649} + 1$; $x^{4423} + x^{1393} + 1$;
9689	$x^{9689} + x^{84} + 1$; $x^{9689} + x^{471} + 1$; $x^{9689} + x^{1836} + 1$; $x^{9689} + x^{2444} + 1$; $x^{9689} + x^{4187} + 1$
9941	$x^{9941} + x^{29} + x^{12} + x^{10} + 1$
11213	$x^{11213} + x^{8218} + x^{6181} + x^{2304} + 1$
19937	$x^{19937} + x^{881} + 1$; $x^{19937} + x^{7083} + 1$; $x^{19937} + x^{9842} + 1$
21701	$x^{21701} + x^{17777} + x^{11796} + x^{5005} + 1$
23209	$x^{23209} + x^{1530} + 1$; $x^{23209} + x^{6619} + 1$; $x^{23209} + x^{9739} + 1$
44497	$x^{44497} + x^{8575} + 1$; $x^{44497} + x^{21034} + 1$

3 Commutative Encryption

Encryption algorithm $\mathrm{EA}_K(M)$, where M is the input message; K is the encryption key, is called commutative, if the ciphertext produced by two consecutive encryptions with keys K_1 and K_2 does not depend on the order of using the keys, i.e.

$$\mathrm{EA}_{K_2}\left(\mathrm{EA}_{K_1}(M)\right) = \mathrm{EA}_{K_1}\left(\mathrm{EA}_{K_2}(M)\right).$$

Analogously to the Pohlig-Hellman algorithm [4] the commutative encryption algorithm can be defined over arbitrary finite field $GF(p^s)$, where $s \geq 1$, having sufficiently large order. The secret key is generated as a pair of two numbers (e, d). The value e is called encryption exponent. It is selected as a random number satisfying the following two conditions: i) $gcd(e, p^s - 1) = 1$; ii) $2^\lambda < e < 2^{\lambda+8}$, where λ is the security parameter defining the $(\lambda/2)$-bit security of the algorithm. For example, if the 80-bit security is required, then it is selected value $\lambda = 160$ (using smaller values λ defines faster encryption process). The value d is called decryption exponent. It is computed as follows:

$$d = e^{-1} \bmod p^s - 1.$$

Encrypting the message M represented as an element of the field $GF(p^s)$ is performed as follows:

$$C = M^e.$$

Decrypting the ciphertext C is performed as follows (performance of the decryption procedure does not depend on the selected value λ, since the size of value d does not depend on λ):

$$M = C^d.$$

The following protocol [13] for transmitting the private message M via public channel uses the commutative encryption algorithm.

1) The sender encrypts the message M with his encryption key $e_s : C_1 = M^{e_s}$ and sends cryptogram C_1 to the receiver.

2) The receiver encrypts the cryptogram C_1 with his encryption key $e_r : C_1 = M^{e_r}$ and sends cryptogram C_2 to the sender.

3) The sender decrypts the cryptogram C_2 with his decryption key $d_s : C_3 = C_2^{d_s}$ and sends cryptogram C_3 to the receiver. Then the receiver recovers the message M as follows $M = C_3^{d_r}$.

Correctness. Proof of the protocol:

$$\begin{aligned} {C_3}^{d_r} &= \left({C_2}^{d_s}\right)^{d_r} \\ &= \left({C_1}^{e_r}\right)^{d_s d_r} \\ &= \left(M^{e_s}\right)^{e_r d_s d_r} \\ &= M^{e_s d_s} \\ &= M. \end{aligned}$$

Protocols like this one are used, for example, in mental poker [13].

Security of the described protocol is based on the DLP in the finite field $GF(p^s)$. Indeed, the values C_2 and C_3 are sent via public channel and the potential attacker can try to solve the equation $C_3 = C_2^{d_s}$ with unknown value d_s. Usually this problem is computationally infeasible, if the order of the field $GF(p^s)$ is sufficiently large. However, if the order ω of the encrypted message as element of $GF(p^s)$ is a small value or ω contains only sufficiently small divisors, then the attacker will be able to solve the equation $C_3 = C_2^{d_s}$ that will give him some information about the key d_s. Let such message be called weak. It is reasonable to use such fields $GF(p^s)$ for which the portion of weak message is negligibly small. The possible case is the use of the ground field $GF(p)$ with characteristic equal to $2q+1$, where q is a prime. For such fields only one weak message exists $M = 2q$. The case that is free from weak message relates to the use of binary fields $GF(2^s)$ such that the value $2^s - 1$ is prime. In this case the order of all possible messages, except $M = 0$ and $M = 1$, have large prime order $\omega = 2^s - 1$.

Thus, commutative encryption algorithm defined over the finite fields $GF(2^s)$ such that their multiplicative group has order equal to sufficiently large Mersenne prime $2^s - 1$ represents an ideal case relatively existence of weak messages. Section 2 presents the appropriate values s and low weight irreducible binary polynomials for defining fast multiplication operation in the mentioned fields.

4 Public-key Encryption

The ElGamal public-key encryption algorithm [3] uses the difficulty of the DLP in the fields $GF(p)$ and can be used for sending a secret message via a public channel to the owner of the public key $y = a^k \bmod p$, where k is private key; p is a large prime; a is a primitive element mod p. The algorithm performs as follows:

1) The sender generates the single-use private key u and computes the single-use public key $R = a^u \bmod p$. Then he computes the single-use secret key $Z = y^u \bmod p$ and encrypts the message M: $C = MZ \bmod p$, where C is the produced ciphertext.

2) Then the values C and R are send to the owner of public key y.

The decryption procedure is performed as follows:

1) Using the value R and private key k the receiver computes the single-use secret key $Z = R^k \bmod p$.

2) Then he decrypts the ciphertext C and obtains the message $M = CZ^{-1} \bmod p$.

Suppose except large prime q some small primes r_i (i=1, 2,..., g) divide the number $p - 1$. Then an adversary can implement some potential known-decrypted-text attack on the ElGamal algorithm that relates to the following scenario. The attacker selects a value $R' < p$ having composite order $\omega' = \prod_{i=1}^{g} r_i$ modulo p, generates a random value $C < p$, and then sends the values R' and C to the owner of the public key y. The receiver computes the value $M' = (CZ'^{-1} \bmod p) = (CR'^{-k} \bmod p)$ that become some way known to the attacker. The last computes the value $Z' = (CM'^{-1} \bmod p) = (R'^{k} \bmod p)$. Then using the baby-step-giant-step algorithm [9] the attacker obtains the value $k' \equiv k \bmod \omega'$. If $\omega' > q > k$, then $k' = k$. If $\omega' < k$, then $k = k' + \eta\omega'$, where η is a natural number such that $\eta < k$. Evidently, finding η is easier than finding the private key k, therefore one can claim that the known decrypted text attack provides computing at least part of the private key. The highest security of the ElGamal algorithm is provided in the case of using the prime values p such that $p = 2q + 1$, where q is also a prime. In the last case the considered attack outputs only one bit of the information about the private key k.

One can propose the following modification of the ElGamal algorithm for which the known-decrypted-text attack outputs no information about k.

Full security against the mentioned known-decrypted-text attack is provided with using binary finite fields $GF(2^s)$, where s is sufficiently large Mersenn exponent and multiplication is defined modulo an irreducible binary polynomial $\pi(x)$ having the degree s. Correspondingly the message M is interpreted as a binary polynomial $M(x)$ of the degree $m < s$ and instead of the integer a in the ElGamal algorithm it is used any binary polynomial $\alpha(x)$, except 0 and 1 (indeed, each of such polynimials $\alpha(x)$ has prime order equal to $2^s - 1$). Thus, the public key is computed as the polynomial $\chi(x) = (\alpha(x))^k \bmod \pi(x)$, where k is the private key, and the public-key encryption is performed as follows:

1) The sender generates at random the single-use private key $u < 2^s - 1$ and computes the single-use public key $\rho(x) = (\alpha(x))^u \bmod \pi(x)$. Then he computes the single-use secret key as the polynomial $\zeta(x) = (\chi(x))^u \bmod \pi(x)$ and encrypts the message $M(x) : C(x) = M(x)\zeta(x) \bmod \pi(x)$, where $C(x)$ is the ciphertext.

2) Then the values $C(x)$ and $\rho(x)$ are sent to the owner of public key y, i.e. to the receiver of the message $M(x)$.

The decryption procedure is performed as follows:

1) Using the single-use public key $\rho(x)$ the receiver computes the value $\zeta(x) = (\rho(x))^k \bmod \pi(x)$.

2) Then he decrypts the ciphertext $C(x)$ and obtains the message $M(x) = C(x)(\zeta(x))^{-1} \bmod \pi(x)$.

5 Conclusions

In this paper it has been proposed a new construction of the zero-knowledge protocol, which is based on the public key agreement scheme. The most simple design of the proposed protocol relates to the case of using binary finite fields $GF(2^s)$ for which the value $2^s - 1$ is a Mersenne prime. It has been also shown that such fields represent significant interest for using them in the commutative and public-key encryption algorithms. Besides, potential application of the Mersenne primes relates to the deniable-encryption schemes [5, 6], especially to the method [12] providing bi-deniability and high performance, however the last represents a topic of individual research though.

Acknowledgments

This work was supported by Government of Russian Federation, Grant 074-U01.

References

[1] R. Crandall and C. Pomerance, *Prime Numbers - A Computational Perspective*, New York: Springer, 2002.

[2] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[3] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469–472, 1985.

[4] M. E. Hellman and S. C. Pohlig, *Exponentiation Cryptographic Apparatus and Method*, US Patent 4, 424, 414, 1984.

[5] M. H. Ibrahim, "A method for obtaining deniable public-key encryption," *International Journal of Network Security*, vol. 8, no. 1, pp. 1–9, 2009.

[6] M. H. Ibrahim, "Receiver-deniable public-key encryption," *International Journal of Network Security*, vol. 8, no. 2, pp. 159–165, 2009.

[7] ISO/IEC, *Information Technology - Security Techniques - Entity Authentication*, Part 5: Mechanisms Using Zero-knowledge Techniques, ISO/IEC 9798-5:2009(E), 2009.

[8] Y. Kurita and M. Matsumoto, "Primitive t-nomials ($t = 3, 5$) over $GF(2)$ whose degree is a Mersenne exponent ≤ 44497," *Mathematics of Computation*, vol. 56, no. 194, pp. 817–821, 1991.

[9] A. J. Menezes, S. A. Vanstone, and P. C. Oorschot, *Handbook of Applied Cryptography*, CRC Press, 1996.

[10] N. A. Moldovyan, "Blind signature protocols from digital signature standards," *International Journal of Network Security*, vol. 13, no. 1, pp. 22–30, 2011.

[11] N. A. Moldovyan and A. A. Moldovyan, "Blind collective signature protocol based on discrete logarithm problem," *International Journal of Network Security*, vol. 11, no. 2, pp. 106–113, 2010.

[12] N. A. Moldovyan and A. A. Moldovyan, "Practical method for bi-deniable public-key encryption," *Quasigroups and Related Systems*, vol. 22, no. 2, pp. 277–282, 2014.

[13] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code*, New York: John Wiley, 1996.

[14] C. P. Schnorr, "Efficient signature generation by smart cards," *Journal of Cryptology*, vol. 4, no. 3, pp. 161–174, 1991.

[15] N. Zierler, "Primitive trinomials whose degree is a Mersenne exponent," *Information and Control*, vol. 15, no. 1, pp. 67–69, 1969.

Dr. Alexander A. Moldovyan is a Professsor at the ITMO University. His research interests include information security and cryptographic protocols. He has authored or co-authored more than 60 inventions and 220 scientific articles, books, and reports. He received his Ph.D. (1996) from the St. Petersburg State Electrotechnical University (SPSEU).

Dr. Nikolay A. Moldovyan is an honored inventor of Russia (2002). His research interests include information security and cryptology. He has authored or co-authored more than 70 inventions and 230 scientific articles, books, and reports. He received his Ph.D. (1981) from the Academy of Sciences of Moldova. Contact him at: nmold@mail.ru.

Andrey N. Berezin received his M.S. degree (2012) in Computer Security from the SPSEU, Russia. He is a Ph.D researcher at the SPSEU. His current research interests include information security and cryptology.