International Journal of Innovation and Applied Studies
ISSN 2028-9324 Vol. 11 No. 1 Apr. 2015, pp. 90-95

http://www.ijias.issr-journals.org/

ZERO-KNOWLEDGE PROTOCOLS BASED ON PUBLIC-ENCRYPTION

Nashwan Ahmed Al-Majmar[1], Dmitriy Nikolaevich Moldovyan[2], and Nikolay Andreevich Moldovyan[2]

[1]Department of Math's and Computers,
Faculty of Science,
Ibb University,
Ibb, Yemen

[2]Department of Automated Systems of Information Processing,
Faculty of Computer technologies and Informatics,
St. Petersburg State Elecrotechnical University "LETI",
St. Petersburg, Russia

ABSTRACT: The paper considers the design of two-step zero-knowledge protocols of two different types: 1) protocols based on the public encryption 2) protocols based on the public key agreement scheme. The novelty of the proposed design relating to the first type of protocols consists in using specified labels that are embedded in the encrypted message. Due to using the labels the proposed design is free of using hash-functions and provides higher performance and cheaper hardware implementation. The paper describes protocols implemented with using El-Gamal, Rabin, and RSA public-encryption algorithms. There are discussed details of the protocol design, which depends on the used public-encryption algorithm. The novelty of the proposed design relating to the second-type protocols consists in using the public key agreement scheme.

KEYWORDS: cryptographic protocol, authentication, public key, secret key, public encryption, discrete logarithm problem, factorization problem

1 INTRODUCTION

Zero-knowledge protocols are used for entity authentication with using public keys [1].They are also called protocols without disclosure of the secret, which represents the private key of the user, who is the owner (who plays the role of the prover in the protocols) of the public key(PK).Interpretation of the notion of zero-knowledge about the secret can be explained as follows. Initially, the owner of the public key provides to everybody and to the potential attacker the value (i.e. the PK) from which the secret can be computed, however this problem is computationally infeasible. While participating in the protocol many times the prover provides no values (no information) that makes the problem of computing the secret a bit easier, he sends to the verifier only values that gives no assistance for computing his private key.

In the known two-step zero-knowledge protocols, for example in the protocols described in the standard ISO / IEC 9798-5: 2009 (E) [2], the prover sends to the verifier (person who verifies authenticity of the prover) only messages that are known to the last. Before sending the message the prover checks that the verifier knows the message, using the hash values computed from the message. While implementing the protocols in hardware, using the hash functions leads to increasing the implementation cost.

This paper proposes a method for reducing the hardware implementation cost of the two-step zero-knowledge protocols. The first proposed design consists in using the labels inserted in the messages that are encrypted using the public key of the prover instead of computing the hash value from the message. The second proposed design is based on using the public key agreement scheme instead of public encryption.

2 TWO-PASS ZERO-KNOWLEDGE PROTOCOLS USING PUBLIC ENCRYPTION

The proposed approach to design tow-pass zero-knowledge protocols is based on the idea of using public key algorithms as the main mechanism of the protocol, provided, for example ,in standard [2]. In the known protocols of this type it is used a public key algorithm *Publ_Encr*. The protocol is implemented as performing the following two steps:

1 The verifier generates some random message *M* and computes the cryptogram $C = Publ\_Encr(M, P)$, where *P* is the public key of the prover, and the hash-function value $H = F_H(M)$, where F_H is some specified hash function. Then the verifier sends the values *C* and *H* to the prover.

2 The prover decrypts the message from the cryptogram, using his private key, and computes the hash-function value from the decrypted message. If the prover obtains the value equal to *H* he concludes the verifier knows the message *M* and sends the decrypted message to the verifier.

Since correct decryption of the ciphertext requires using the private key, the verifier concludes the prover is valid.

In the proposed implementation of the two-pass zero-knowledge protocols it is used a simpler mechanism for checking fact that the verifier knows the value obtained by the prover with the decryption procedure. This mechanism consists in imbedding some pre-specified labels μ in the message encrypted by the verifier. The presence of such labels in the decrypted message convinces the prover that the cryptogram was formed correctly, i.e. as result of the public encryption. The last means the verifier knows the result of decryption, therefore, while obtaining the prover's responce, the verifier gets no information about the private key.

2.1 PROTOCOL BASED ON THE RSA PUBLIC ENCRYPTION

The design of zero-knowledge protocol using the public key algorithm RSA [3] is described as follows. While computing the public key one selects two large strong primes *r* and *q* [4], computes the product $n = rq$ and the value of the generalized Euler function $L(n)$ that is equal to the least common multiple of the numbers $r-1$ and $q-1$. After that it is generated a random 32-bit number *e* which is relatively prime to $L(n)$ and it is computed the number *d* satisfying the condition:

$$ed \equiv 1 \bmod L(n)$$

The pair of values *n* and *e* is a public key and the value *d* is a private key. The procedure of public-encryption of the message $M < n$ is described as follows:

$$C = M^e \bmod n$$

where *C* is the cryptogram (ciphertext). Decryption of the cryptogram *C* is described with the following formula:

$$M = C^d \bmod n$$

The correctness of the decryption procedure can be easily proved using the generalized Euler's theorem, according to which for any number *M* that is relatively prime to *n*, the following relation is true:

$$M^{L(n)} \equiv 1 \bmod n$$

To provide the 80-bit (128-bit) security the size of the primes *p* and *q* is to be selected equal to 512 bits (1250 bits). The two-round zero-knowledge protocol based on the RSA public encryption uses a 256-bit label μ. For example, one can take the following 256-bit prime number:

μ=109366802940193868629366966069390547386166074451532541882671671583165839011751.

The proposed protocol is described by the following steps:

1.The verifier generates a random message *M* with size $|M|$, satisfying the condition $2|n|/3 < |M| < |n|-|\mu|$. Then he encrypts the message *M* to which the label μ is attached, i.e. he encrypts the bit string $M||\mu$ using prover's public key (*n, e*), i.e. with using the formula $C = (M||\mu)^e \bmod n$, and sends the value *C* to the prover as request, on which he expects a response.

2. The prover decrypts the cryptogram *C* using his personal secret key *d* and formula $M'||\mu' = C^d \bmod n$, where μ' is a bit string representing 256 of the right bits of the value $C^d \bmod n$. Then the prover compares the values $\mu'$and μ. If $\mu' = \mu$, then the prover send the value M' to the verifier as its response to the received request. Otherwise, the prover sends the response "Invalid request".

The verifier compares the values M' and M. If $M' = M$, then the verifier concludes that the prover is authentic.

In the described protocol it is important to use a pre-specified label μ, thereby provided opportunity for the prover to make sure that the value of the answer M' is already known for the verifier. The fact of that, it is already known for the verifier, means, that the verifier does not try to get a signature of the verifier to a certain message, using a mechanism of blind signature [5],[6], and does not attempt to carry out the attack using adaptively selected ciphertext (adaptive chosen ciphertext attack) described in [7], or some other attack.

2.2 PROTOCOL BASED ON EL-GAMAL PUBLIC ENCRYPTION ALGORITHM

Let us consider the implementation of the protocol using the El-Gamal public key algorithm [8]. The method for public encryption by El-Gamal uses the public having the form $y = \alpha^x \bmod p$, where p is some specified large prime (having size 1024 bits or more) such that the number $p - 1$ has a prime divisor (having size 160 bits or more) and α is some specified primitive element modulo p.

Actually, the El-Gamal public key algorithm is a hybrid cryptosystem, in which the secret keys are distributed accordingly to the Diffie-Hellman protocol [9], and the message encryption is done as multiplying (modulo p) the message on the single-use secret key. Encrypting a message T, which is to be send to the owner of the public key y, is performed as follows:

1.Generate a random number k which essentially is a single-use (one-time) secret key.

2. Calculate the number $R = \alpha^k \bmod p$ which essentially is a single-use (one-time) public key of the sender.

3. Calculate the single-use (one-time) shared secret key $Q = y^k \bmod p$, where y is the receiver's public key.

4.Encrypt the message M by multiplying the messages on the single-use secret key: $C = QM \bmod p$.

5. Send to the receiver the cryptogram representing the pair of numbers(R, C).

Suppose the procedure of the El-Gamal public encryption using the public key y be denoted as *Gamal_Encr*(M, y). Receiver of the cryptogram (R, C) performs decryption procedure *Gamal_Decr*(R, C, x),using his private key x as follows:

1. Calculate the single-use shared secret key $Q = R^x \bmod p$.

2. Using the extended Euclidean algorithm compute the value Q^{-1} that is inverse to the value Q modulo p.

3. Decrypt the cryptogram C by multiplying C on the integer Q^{-1}: $M = CQ^{-1} \bmod p$.

El-Gamal encryption algorithm uses random values, i.e.it implements a probabilistic encryption procedure, in which the same message corresponds to a set of cryptograms, all of which give the same output value M after the decryption procedure is completed.

The proposed authentication protocol with zero-knowledge based on the El-Gamal algorithm uses a 128-bit label μ. For example, one can take the following 128-bit prime number:

$$\mu = 296679166210822344381018575511334242983.$$

The protocol is described as follows:

1. The verifier generates a random message M having size $|M|$ and satisfying the condition $|M| < |p| - |\mu|$. Then, using the El-Gamal public encryption algorithm and prover's public key y, he encrypts the message M to which the label μ is attached, i.e. he encrypts bit string $M||\mu$ in accordance with the formula $(R, C) = Gamal\_Encr(M||\mu, y)$. After that he sends to the prover the pair (R, C) as its request on which the prover must answer.

2. The prover decrypts the cryptogram (R, C) using his private key x: $M'||\mu' = Gamal\_Decr(R, C, x)$, where μ' is a bit string represented by lower 128 bits of the decrypted value $M'||\mu'$. Then prover compares the values μ' and μ. If $\mu' = \mu$, then the prover send the value M' to the verifier as his response to the received request. Otherwise, the prover sends the response "Invalid request."

If the verifier does not intend to obtain information about prover's private key, then he correctly performs the first step of the protocol using the specified label μ, attached to the message M and receives as response to his request the value $M' = M$, i.e. the verifier receives the value already known for him and he concludes the prover is authentic. The equality $\mu' = \mu$

holds for a randomly selected request send to the prover only with negligible probability (equal to $2^{-|\mu|}$), i.e. only with negligible probability the prover can send a reply on an incorrect request sent by the verifier.

2.3 PROTOCOL BASED ON RABIN PUBLIC-ENCRYPTION ALGORITHM

In public key algorithm by Rabin [10] it is used calculations modulo a composite number $n = pq$ that is difficult for factoring and used as public key. The pair of strong primes p and q that satisfy conditions $p \equiv 3 \bmod 4$ and $q \equiv 3 \bmod 4$ represents the private key. Two last conditions provide for the owner of the public key n possibility to compute easily square roots modulo n. The last operation represents the deciphering procedure. The encryption of the message $M < n$ is performed as squaring number M modulo n:

$$C = M^2 \bmod n$$

The decryption procedure is performed by finding square roots from the cryptogram C modulo n as follows. Preliminary there are computed roots from C modulo p and modulo q:

$$m_{p_1} = C^{\frac{p+1}{4}} \bmod p, \quad m_{p_2} = p - m_{p_1} = p - C^{\frac{p+1}{4}} \bmod p,$$

$$m_{q_1} = C^{\frac{q+1}{4}} \bmod q, \quad m_{q_2} = q - m_{q_1} = q - C^{\frac{q+1}{4}} \bmod q.$$

Then there are computed four roots from C modulo n using the following four formulas:

$$M_1 = \left(m_{p_1} a + m_{q_1} b \right) \bmod n,$$

$$M_2 = \left(m_{p_1} a + m_{q_2} b \right) \bmod n,$$

$$M_3 = \left(m_{p_2} a + m_{q_1} b \right) \bmod n,$$

$$M_4 = \left(m_{p_2} a + m_{q_2} b \right) \bmod n,$$

where $a = q\,(q^{-1} \bmod p)$ and $b = p\,(p^{-1} \bmod q)$.

Like in the case of the RSA algorithm, to provide the 80-bit (128-bit) security one can select the size of the primes p and q equal to 512 bits (1250 bits). In the two-round zero-knowledge protocol based on the Rabin public encryption algorithm one can use, for example, the 256-bit label μ mentioned in *Subsection 1.1*. The protocol is described as follows:

1. The verifier generates a random message M having size $|M|$ that satisfies the condition $2|n|/3 < |M| < |n| - |\mu|$ and encrypts the message M to which some specified label μ is attached, i.e. the verifier encrypts the value $M||\mu$ in accordance with the formula $C = (M||\mu)^2 \bmod n$. Then he sends the value C to the prover as his request on which he expects a response from the prover.

2. The prover decrypts cryptogram C using his private key (p, q) by calculating the four square roots from C: M_1, M_2, M_3 and M_4. Each of the last values he represents as $M_i' || \mu_i'$, where the bit string μ' is given by the lower 2565 bits of the ith root M_i' (i= 1,2,3,4). Then he checks, whether the equality $\mu_i' = \mu$ holds for one of the four values M_i. If the equality takes place, then the prover sends the respective value M_i' to the verifier as his response to the received request. If $\mu_i' \neq \mu$ for i = 1,2,3,4, then the prover sends the response "Invalid request".

Using pre-specified labels allows the prover to identify the original message, so he sends to the verifier he value that is known for the verifier. This provides zero leakage of the information about the private key while the prover sends his response to the verifier. By analogy with the protocol described in this subsection he can design the two-step zero-knowledge protocols using the provably secure public encryption algorithm described in [11] for which the cryptogram is deciphered in three different messages M_i (i= 1,2,3).

3 TWO-PASS ZERO-KNOWLEDGE PROTOCOLS USING THE PUBLIC KEY AGREEMENT SCHEME

The idea of using the public key agreement schemes to develop a zero-knowledge protocol relates to the possibility of two users' generating a common secrete value with the help of exchanging their public keys. The verifier is to use the public key of the prover and generate his single-use public key to implement the public key agreement scheme each time when validity of the prover has to be performed with the zero-knowledge protocol. The public key agreement scheme is implemented in frame of the zero-knowledge protocol for verifier's pre-computing the prover's response. For example, one can use the Diffie–Hellman cryptoscheme [9] in which the public key y is generated as follows:

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

where p is a sufficiently large prime such that some another large prime q divides the number $p-1$; α is a primitive element modulo p; k is a randomly selected number ($k<p-1$) that serves as private key. The public key agreement is performed as follows. One user selects a private key k and computes his public key y. Another user generates his private key $u<p-1$ and computes his public key $R=\alpha^u \bmod p$. After exchange of the public keys the first user computes the common secret value as follows:

$$Z = R^k \bmod p$$

The second user computes the common secret using the formula:

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

Any other person is not able to compute Z until he solves the discrete logarithm problem and gets the value k or the value u from the known value y or R. Suppose the first user is the prover in the zero-knowledge protocol, i.e. is the person to be authenticated by the second user that plays the role of verifier. Suppose also there is used some specified hash function $h(*)$ and the verifier has been provided with a trusted copy of prover's public key y.

The proposed zero-knowledge protocol based on the public key agreement scheme is implemented in the following two steps:

1. The verifier generates a random single-use key $u<p-1$ and computes the single-use public key $R=\alpha^u \bmod p$. Then he computes the single-use common secret $Z=y^u \bmod p$ and the hash-function value $H=h(Z)$ and sends to the prover the pair of numbers (R,H) as his request.

2. After receiving the request (R,H) and using the private key k the prover computes the values $Z'=R^k \bmod p$ and the hash-function value $H'=h(Z')$. Then he compares the values H' and H. If they are equal, then the prover sends to the verifier the value Z' that is claimant's response. If $H' \neq H$, the prover sends the message "Incorrect request".

The verifier compares the values Z' and Z. If $Z'=Z$, then the verifier concludes the prover knows the private key relating to the public key y, otherwise he concludes the prover is not valid. At step 2 it is important for the prover to check that equality $H'=H$ holds, since in the last case the prover is convinced the verifier has already computed the value Z and zero information about the secret key is passed to the verifier when the last receives the prover's response.

4 DISCUSSION AND CONCLUSION

Zero-knowledge protocols relates to so called provably secure cryptoschemes, therefore their security is defined by the difficulty of the computational problem put into their base. Depending on the required security level of the entity authentication procedure one should only select the respective size of the prime modulo p (of the composite number n) for the case of zero-knowledge protocols based on difficulty of the discrete logarithm (factoring) problem. This remark can be attributed to the known and to the proposed two-step zero-knowledge protocols, therefore the discussion is focused on the performance and hardware implementation cost of the protocols.

The first type of the proposed protocols are based on using the public encryption algorithm. They differ from the known two-pass zero-knowledge protocols by using pre-specified labels concatenated to the encrypted messages instead of computing the hash-function values from the messages. Therefore protocols became a bit faster since they are free from computing hash-function values (while both the hardware and the software implementation). Probably for majority of applications this is not a significant advantage. However, while hardware implementation the cost is significantly reduced since for implementing the proposed protocols there is no need to spend hardware resources for implementation of the hash-function algorithm.

The second type of the proposed protocols are also only a bit faster than the known two-pass zero-knowledge protocols based on the difficulty of discrete logarithm (the lasts require performing two additional multiplications mod *p* and one inversion modulo *p*). The advantage of the proposed design of the second type relates to easier implementation while the protocols are implemented on the base of the difficulty of the discrete logarithm problem on elliptic curves or on the base of difficulty of the discrete logarithm in a hidden cyclic subgroup of finite non-commutative groups [12],[13].

Thus, in this paper there have been proposed two new designs of two-pass zero-knowledge protocols that have some advantages while practical application.

REFERENCES

[1] Fiat A., Shamir A. How to prove yourself: Practical solutions to identification and signature problems // Advances in cryptology – CRYPTO'86, Springer-Verlag LNCS, 1987. Vol. 263. P. 186–194

[2] ISO/IEC 9798-5:2009(E) «Information technology — Security techniques — Entity authentication — Part 5: Mechanisms using zero-knowledge techniques».

[3] Rivest R., Shamir A., Adleman A. A method for Obtaining Digital Signatures and Public-Key Cryptosystems // Communication of the ACM. 1978. Vol. 21. N. 2. P. 120–126.

[4] Gordon J. Strong primes are easy to find // Advances in cryptology – EUROCRYPT'84. Springer-Verlag LNCS. 1985. Vol. 209. P. 216–223.

[5] Chaum D. Blind Signatures for Untraceable Payments // Advances in Cryptology: Proc. of CRYPTO'82. Plenum Press, 1983. P. 199–203.

[6] Pieprzyk J., Hardjono Th., Seberry J. Fundumentals of Computer Security. Springer-verlag. Berlin, 2003. – 677 p.

[7] Bleichenbacher D. Chosen cipher text attacks against protocols based on the RSA encryption standard PKCS #1 // Advances in Cryptology – CRYPTO'98. Springer-Verlag LNCS. 1998. Vol. 1462. P. 1–12.

[8] El-Gamal T. A public key cryptosystem and a signature scheme based on discrete logarithms // IEEE Transactions on Information Theory. 1985. Vol. IT-31. No. 4. P.469–472

[9] Diffie W., Hellman M.E. New Directions in Cryptography // IEEE Transactions on Information Theory. 1976, Vol. IT-22. P. 644–654.

[10] Rabin M.O. Digitalized signatures and public key functions as intractable as factorization. – Technical report MIT/SK/TR-212, MIT Laboratory for Computer Science, 1979.

[11] Moldovyan N.A., Moldovyan A.A. Class of Provably Secure Information Authentication Systems // 4th Int. Conference on Mathematical Methods, Models, and Architectures for Computer Network Security, MMM-ANCS'07 Proc. September 13-15, 2007 / Springer Verlag CCIS. 2007. Vol. 1, P.147–152.

[12] Moldovyan D.N., Moldovyan N.A. A New Hard Problem over Non-Commutative Finite Groups for Cryptographic Protocols // 5thInt. Conference on Mathematical Methods, Models, and Architectures for Computer Network Security, MMM-ANCS 2010 Proceedings. St.Petersburg, September 8-11, 2010 / Springer Verlag Lecture Notes in Computer Science. 2010. Vol. 6258. P. 183–194.

[13] Moldovyan D.N. Non-Commutative Finite Groups as Primitive of Public-Key Cryptoschemes // Quasi groups and Related Systems. 2010. Vol. 18. P. 165–176.c