No-Key Protocol for Deniable Encryption

Nam Hai Nguyen[1], Nikolay Andreevich Moldovyan[2], Alexei Victorovich Shcherbacov[3], Hieu Minh Nguyen[1](✉), and Duc Tam Nguyen[1]

[1] Academy of Cryptography Techniques, Hanoi, Vietnam
{hainnvn61, hieuminhmta, nguyenductamkma}@gmail.com

[2] St.Petersburg Institute for Informatics and Automation of Russian Academy of Sciences, 199178 St.Petersburg, Russia
nmold@mail.ru

[3] Institute of Mathematics and Computer Science of Academy of Sciences of Moldova, Academiei str. 5, 2028 Chishinau, Moldova
scerb@math.md

Abstract. There is proposed a new method for deniable encryption based on commutative transformations. The method has been used to design the deniable encryption protocol resistant to the passive coercive attacks, which uses no pre-shared secret keys and no pre-exchanged public keys. The protocol begins with the stage at which the sender and receiver exchange their single-use public keys and compute the single-use shared secret key. Then, it is performed pseudo-probabilistic three-pass protocol with simultaneous commutative encryption of the fake and secret messages. Resistance of the proposed protocol to coercive attacks is provided by its computational indistinguishability from probabilistic no-key three-pass protocol used to send securely the fake message. To perform commutative encryption, it used exponentiation cipher. To provide security against active coercer, the protocol is to be complemented with procedure for authenticating the sent messages.

1 Introduction

Protocols and algorithms for deniable encryption (DE) allow one to solve a number of specific practical problems for providing information security of the information technologies [1,2]. The DE cryptoschemes are characterized in their resistance to the coercive attacks in the model of which it is supposed that the coercive attacker (coercer) intercepts ciphertext transmitted via an open communication channel and has possibility to force sender and receiver of the message to disclose the plaintext and the used secret keys, including private keys in the case of using the public key encryption algorithms. In the literature, the public key DE protocols [3] and the shared-key DE ones [4] are described. Recently, it has been proposed DE protocol based on using commutative encryption algorithm and shared secret key [5]. Commutative encryption procedures (commutative ciphers) are very interesting for practical application in the case

V. Bhateja et al. (eds.), *Information Systems Design and Intelligent Applications,* Advances in Intelligent Systems and Computing 672,
https://doi.org/10.1007/978-981-10-7512-4_10

of passive potential attacks, since they can be put into the base of no-key encryption protocols that permit one to transmit a secret message via a public channel without using public and secret keys shared by the receiver and the sender.

To provide resistance to passive attacks, the no-key encryption protocol should be based on some commutative cipher that is secure to the known-plaintext attack. The Pohlig–Hellman exponentiation cipher [6] represents a commutative encryption algorithm that satisfies the indicated requirement. Usually, when considering the DE protocols, it estimated their resistance to potential coercive attacks implemented by passive attacker. For the first time, the problem of providing security of the DE protocols to active coercive attacks had been discussed in papers [7,8]. To provide security to attacks of active coercer, it had been proposed to include in the DE protocols procedure of mutual authentication of the sender and receiver. Remaining in the framework of the model of the passive coercive attacks, it represents theoretical and practical interest to construct no-key DE protocol.

This paper first provides a method for no-key DE and describes a protocol that implements this method. The designed protocol begins with the execution of the public key agreement that provides the sender and receiver the secret message with shared single-use secret value Z. Then, it is implemented simultaneous encryption of the secret T and fake M messages, the encryption being computationally indistinguishable by the ciphertexts from the probabilistic no-key encryption of the message M. The value Z is used as a parameter of the probabilistic no-key encryption protocol on which the probabilistic no-key encryption depends. In the case when both the sender and the receiver are coerced, they disclose the fake message M, the value Z, and local keys used for commutative encryption of the message M and intermediate ciphertexts relating to M. They also declare and insist that during the communication session, they used the probabilistic no-key encryption protocol in order to send securely the massage M. Using the disclosed values, it is computationally impossible for the coercer to insist reasonably that the sender and the receiver used the no-key DE method.

In this paper, Sect. 2 presents an overview of the used cryptosystems. Section 3 presents the no-key deniable encryption method. Section 4 describes the no-key probabilistic encryption protocol. Section 5 presents the no-key deniable encryption protocol that implements secure transmission of the secret message T via a public channel, which is resistant to passive coercive attacks. Section 6 summarizes the results of the paper.

2 Used Cryptosystems

The proposed method for no-key DE includes as its three basic components the following cryptographic primitives: the Diffie–Hellman public key agreement protocol, Pohlig–Hellman commutative encryption algorithm, and no-key encryption protocol.

In the Diffie–Hellman protocol [9] it used a sufficiently large prime number p (having size, e.g., not less than 2,464 bits, which provides the 128-bit security), such that number (p - 1) contains a large prime divisor r (e.g., having size at least 256 bits), and the number α that is a primitive element modulo p. Each user chooses a private key as a random number x $(0 < x < p - 1)$ having size more than 256 bits, and computes his public key y in accordance with the formula: $y = \alpha^x \bmod p$.

Then, the owner of the public key registers his public key in a specially created certification center, called certificate authority (CA). All public keys are placed in a public directory which is signed by a CA in order to avoid possible attacks with substituting public keys or imposing of the false public keys. If two users A and B want to establish secret communication, they proceed as follows. User A takes public key of user B out a public key directory and computes the shared secret Z_{AB}:

$$Z_{AB} \equiv y_B^{x_A} \equiv (\alpha^{x_B})^{x_A} \equiv \alpha^{x_B x_A} \bmod p \tag{1}$$

where y_B is user B's public key and x_A is user A's private key. The users have no need to transmit the shared secret key Z_{AB} via a communication channel, since calculates the value Z_{AB} by a similar formula:

$$Z_{AB} \equiv y_A^{x_B} \equiv (\alpha^{x_A})^{x_B} \equiv \alpha^{x_B x_A} \bmod p \tag{2}$$

where y_A is user A's public key; x_B is user B's private key. It is assumed that a potential attacker knows the values of $y_B = \alpha^{x_B} \bmod p$ and $y_A = \alpha^{x_A} \bmod p$, available in a public directory or digital certificates (signed by a CA) that are exchanged between users A and B via a public channel. However, in order to calculate the value Z_{AB}, the attacker has to solve computationally difficult problem of the discrete logarithm. Shared secret Z_{AB} can be used by users to encrypt the session secret key with which secret message can be encrypted using some symmetric encryption algorithm.

The Pohlig–Hellman commutative encryption algorithm [6,10] represents performing operation of raising the plaintext to a secret degree e (minimum length of the value e is equal to 256 bits) modulo a large prime p (requirements to the prime p coincide with the requirements to the modulus p in the previous cryptoscheme). Encryption and decryption are performed as raising to different degrees e and d, respectively. Encryption of the message $M < p$ is described as computation of the ciphertext C using the formula $C = M^e \bmod p$. Decryption is represented by the formula $M = C^d \bmod p = M^{ed} \bmod p$. The correctness of the decryption is provided with the condition $ed = 1 \bmod (p - 1)$ which should be implemented while generating the secret key (e, d). To be able to fulfill the last condition, it should be selected the number e that is relatively prime to the number $(p - 1)$. Then using the extended Euclidean algorithm, it is easy to compute the respective inverse value $d = e^{-1} \bmod (p - 1)$.

Thus, the Pohlig–Hellman exponential cipher represents the commutative encryption function, the encryption procedure $E_K(M)$ of which is described by the formula:

$$C = E_K(M) = M^e \bmod p \tag{3}$$

The corresponding decryption procedure D_K is as follows:

$$M = D_K(C) = C^d \bmod p \tag{4}$$

where $D_K = E_K^{-1}$ and the encryption key $K = (e, d)$. Security of the Pohlig–Hellman algorithm to known-plaintext attack is as high as computational difficulty of the discrete logarithm problem.

No-key encryption protocol uses some persistent commutative encryption function $E_K(M)$, where M is the input message and K is the encryption key, i.e., the function for which the following equality holds:

$$E_{K_A}(E_{K_B}(M)) = E_{K_B}(E_{K_A}(M)) \tag{5}$$

where K_A and $K_B (K_B \neq K_A)$ are different encryption keys. The property of commutativity of the encryption function $E_K(M)$ is exploited in Shamir's no-key protocol (also called Shamir's three-pass protocol) that includes the following three steps [10]:

1. Sender of the message M generates a random key K_A and calculates the ciphertext $C_1 = E_{K_A}(M)$. Then, he sends C_1 to the receiver via an open channel.
2. Receiver generates a random key K_B, encrypts the ciphertext C_1 with the key K_B as follows $C_2 = E_{K_B}(C_1) = E_{K_B}(E_{K_A}(M))$, and sends C_2 to the sender.
3. Sender, using decryption procedure $D = E^{-1}$, calculates the ciphertext $C_3 = D_{K_A}(C_2) = D_{K_A}(E_{K_B}(E_{K_A}(M))) = D_{K_A}(E_{K_A}(E_{K_B}(M))) = E_{K_B}(M)$ and sends C_3 to the receiver of the message M.

Using the received ciphertext C_3, the receiver recovers message M according to the formula $M = D_{K_B}(C_3) = D_{K_B}(E_{K_B}(M)) = M$.

In this protocol used encryption keys K_A and K_B are local parameters of commutative transformations; therefore, one can call them local keys. Since the sender and the receiver do not use any shared key, the protocol is called the no-key protocol.

3 No-Key Deniable Encryption Method

Suppose some remote user A wishes to send a secret message $T < p$ to a remote user B, using the no-key encryption protocol, so that they can securely open the local keys K_A and K_B, if passive coercer intercepting ciphertexts C_1, C_2, and C_3 will attack them. In this case, conserving secrecy of the message T is possible, if the ciphertexts C_1, C_2, and C_3 are produced as simultaneous ciphering two different messages, the message T and some fake message M. Besides, the encryption process should look like probabilistic ciphering of the fake message M in the frame of no-key protocol that uses probabilistic commutative encryption function. It is proposed that the encryption method can include the following steps:

1. In accordance with the Diffie–Hellman protocol, the sender and the receiver generate session (single-use) public keys and exchange with them. Then they compute the single-use (session) shared secret key Z that is actual only in the current communication session.
2. User A generates a fake message $M < p$.
3. Users A and B perform the no-key encryption protocol using a commutative encryption function allowing to perform simultaneous ciphering the messages M and T. During the commutative ciphering, each of the users applies two different local keys for encrypting messages T and M.

At step three, it is to be used the encryption function that is computationally indistinguishable from the probabilistic commutative encryption function applied to the fake message. In other words, the ciphertexts C_1, C_2, and C_3 produced with the commutative encryption function for ciphering simultaneously the messages M and T could be potentially produced with the probabilistic commutative encryption function applied to the fake message, the probabilistic encryption process being dependent on the single-use shared key. Since the key Z is produced during the communication session without using any pre-agreed keys (secret or public), the communication protocol can be attributed to the class of no-key protocols. Possibility to connect the ciphertexts C_1, C_2, and C_3 with the probabilistic no-key protocol allows users, in case of the passive coercive attacks, to disclose only local keys used for transformation of the fake message M. To catch the users that they are cheating should be computationally impossible for the passive coercer while using the disclosed local keys, the single-use private keys, the single-use shared secret Z, and fake message M. Section IV introduces appropriate probabilistic no-key protocol, and in Section V, it proposed no-key DE protocol satisfying the last requirement.

4 No-Key Probabilistic Encryption Protocol

Two approaches can be used to provide no-key encryption protocols resistance to attacks based on chosen source message. The first approach is to select a prime modulus p, such that the number of $(p-1)/2$ is prime. The second approach is to embed probabilistic mechanisms in the original protocol. Thus, the users have reasonable motivation to use the probabilistic no-key protocol and this is significant for assigning probabilistic protocols with the DE protocols. Let us consider mechanism for providing security to chosen plaintext attacks.

When ciphering procedure depends on randomly selected values and the single-use shared key Z, a potential attacker performing the chosen plaintext attack is not able to eliminate the influence of random parameter on the produced ciphertext; therefore, his attack is inefficient, when encryption procedure is properly composed. If the key Z is produced during the communication, then a common assumption that prior to the execution of the no-key encryption protocol the sender and the receive do not share any secret values (keys) and have no public keys registered in the CA. (Otherwise, there is no need to use a no-key encryption protocol to send a secret message via a public channel, since to solve the problem one can use symmetric or public encryption.) Thus, applying the single-use shared keys does not contradict the notion of the no-key encryption.

Using exchange of the single-use public keys and computation of the single-use shared secret value, it has been designed the following protocol implementing the probabilistic no-key encryption of the message $M < p$:

1. User A generates a random value $k_A < (p - 1)$, which plays the role of his private single-use key, computes his public single-use key $R_A = \alpha^{k_A} \bmod p$, and sends the value R_A to the user B.
2. User B generates a random value $k_B < (p - 1)$ as his private single-use key, computes his public single-use key $R_B = \alpha^{k_B} \bmod p$, and sends the value R_B to the user A.
3. User A generates his local key $K_A = (e_A, d_A)$, where $d_A = e_A^{-1} \bmod (p - 1)$ calculates the single-use shared secret $Z = R_B^{k_A} \bmod p$, generates a random value ρ, and computes the ciphertext $C_1 = (C'{}_1, C''{}_1)$ as the solution of the following system of linear equations with the unknown C_1' and C_1'':

$$\begin{cases} C_1' + C_1'' = \rho \bmod p \\ C_1' + ZC_1'' = M^{e_A} \bmod p \end{cases} \tag{6}$$

   Then, user A sends the ciphertext C_1 to the user B.

1. User B generates his local key $K_B = (e_B, d_B)$, where $d_B = e_B^{-1} \bmod (p - 1)$ calculates the single-use shared secret $Z = R_A^{k_B} \bmod p$ and the value $S_1 = M^{e_A} \bmod p = (C_1', ZC_1'') \bmod p$, generates a random value ρ', and calculates the ciphertext $C_2 = (C_2', C_2'')$ as the solution of the following system of linear equations with the unknowns C_2' and C_2'':

$$\begin{cases} C_2' + C_2'' = \rho' \bmod p \\ C_2' + ZC_2'' = S_1^{e_B} \bmod p \end{cases} \tag{7}$$

   Then, user B sends the ciphertext C_2 to the user A.
2. User A generates a random value ρ'' and calculates value $S_2 \equiv S_1^{e_B} \equiv (C_2' + ZC_2'') \bmod p$ and ciphertext $C_3 = (C_3', C_3'')$ as solution of the following system of equations with the unknowns C_3' and C_3'':

$$\begin{cases} C_3' + C_3'' = \rho'' \bmod p \\ C_3' + ZC_3'' = S_2^{e_A} \bmod p \end{cases} \tag{8}$$

   Then, user A sends the ciphertext C_3 to the user B.
   Having received the value C_3, user B computes the message M as follows: $M = (C_3' + ZC_3'')^{d_B} \bmod p$.

5 No-Key Deniable Encryption

Using general construction scheme of the no-key deniable encryption protocol described in Sect. 3 and probabilistic no-key encryption protocol presented in Sect. 4, it is easy to write down the following protocol that implements secure transmission of the secret message $T < p$ via a public channel, which is resistant to passive coercive attacks:

1. Sender of the message T generates randomly his single-use private key k_A, calculates his single-use public key $R_A = \alpha^{k_A} \bmod p$, and sends R_A to the receiver.
2. Receiver generates randomly his single-use private key k_B, calculates his single-use public key $R_B = \alpha^{k_B} \bmod p$, and sends R_B to the user A.
3. Sender generates his local keys $K_A = (e_A, d_A)$, where $d_A = e_A^{-1} \bmod (p-1)$, and $Q_A = (\varepsilon_A, \delta_A)$, where $\delta_A = \varepsilon_A^{-1} \bmod (p-1)$, calculates the single-use shared secret $Z = R_B^{k_A} \bmod p$, forms a fake message $M < p$, and calculates the ciphertext $C_1 = (C_1', C_1'')$ as a solution of the following system of equations with the unknowns C_1' and C_1'':

$$\begin{cases} C_1' + Z^2 C_1'' = T^{\varepsilon_A} \bmod p \\ C_1' + Z C_1'' = M^{e_A} \bmod p \end{cases} \tag{9}$$

   Then, the sender sends the ciphertext C_1 to the receiver.
4. The receiver generates his local keys $K_B = (e_B, d_B)$, where $d_B = e_B^{-1} \bmod (p-1)$, and $Q_B = (\varepsilon_B, \delta_B)$, where $\delta_B = \varepsilon_B^{-1} \bmod (p-1)$, calculates the single-use shared secret $Z = R_A^{k_B} \bmod p$, and calculates the values $S_1 \equiv M^{e_A} \equiv (C_1' + ZC_1'') \bmod p$ and $U_1 \equiv T^{e_A} \equiv (C_1' + Z^2 C_1'') \bmod p$ and the ciphertext $C_2 = (C_2', C_2'')$ as solution of the following system of equations with the unknowns C_2' and C_2'':

$$\begin{cases} C_2' + Z^2 C_2'' = U_1^{e_B} \bmod p \\ C_2' + Z C_2'' = S_1^{e_B} \bmod p \end{cases} \tag{10}$$

   Then, the receiver sends the ciphertext C_2 to the sender.
5. The sender calculates the values $S_2 \equiv S_1^{e_B} \equiv (C_2' + ZC_2'') \bmod p$ and $U_2 \equiv U_1^{e_B} \equiv (C_2' + Z^2 C_2'') \bmod p$ and ciphertext $C_3 = (C_3', C_3'')$ as solution of the following system of equations with the unknowns C_3' and C_3'':

$$\begin{cases} C_3' + Z^2 C_3'' = U_2^{\delta_A} \bmod p \\ C_3' + Z C_3'' = S_2^{e_A} \bmod p \end{cases} \tag{11}$$

   Then, the sender sends the value C_3 to the receiver.

After receiving the ciphertext C_3, the receiver computes the message T:

$$T = (C_3' + Z^2 C_3'')^{\delta_B} \bmod p \tag{12}$$

If necessary (in the case of coercive attack), the receiver can also be calculated fake message M as follows:

$$M = (C_3' + ZC_3'')^{d_B} \bmod p \tag{13}$$

The proof of the correctness of the protocol:
Recovery of the secret message:

$$\left(C_3' + Z^2 C_3''\right)^{\delta_B} \equiv \left(U_2^{\delta_A}\right)^{\delta_B} \equiv \left(U_1^{\varepsilon_B}\right)^{\delta_A \delta_B} \equiv \left(T^{\varepsilon_A}\right)^{\varepsilon_B \delta_A \delta_B} \equiv T \bmod p \tag{14}$$

Recovery of the fake message:

$$(C_3' + ZC_3'')^{d_B} \equiv \left(S_2^{d_A}\right)^{d_B} \equiv (S_1^{e_B})^{d_A d_B} \equiv (M^{e_A})^{e_B e_A d_B} \equiv M \bmod p \qquad (15)$$

When being coerced by a passive attacker, the sender and receiver of the message disclose the fake message M and the keys k_A, R_A, k_B, R_B, Z, (e_A, d_A), and (e_B, d_B). They also say that for securely sending the message M, they used a probabilistic no-key encryption protocol. Since the intercepted by the attacker by procedures specified by the probabilistic no-key encryption protocol associated with no-key deniable encryption protocol, the attacker has the following two possibilities: (i) to agree with the users and (ii) to prove the ciphertexts were produced with the no-key DE protocol. However, the second possibility is computationally infeasible, since to show the difference between the values $\rho_i = (C_i' + C_i'') \bmod p (i = 1, 2, 3)$ and random values the coercer has to compute one of the local keys Q_A or Q_B and to recover the message T. Computing one of the local keys Q_A or Q_B is connected with solving the problem of finding the discrete logarithm modulo p. The last is selected so that computing discrete logarithm is computationally impracticable (see Sect. 2).

In comparison with the known public key DE protocols [11,13] in which the message is encrypted consecutively bit by bit (each bit is sent in the form of 1024-bit pseudorandom number) in the proposed protocols, the message is transformed as a single data block that provides significantly higher performance.

6 Conclusion

Applying commutative encryption algorithm, it has been proposed a method for no-key DE implemented as simultaneous ciphering two messages, secret and fake ones, which is based on public agreement of the single-use shared key with exchange of the single-use public keys of the participants of communication protocol. An important point of the method is fulfillment of the requirement of computationally indistinguishable from probabilistic no-key encryption protocol. To implement the method as an practical no-key DE protocol, it designed a probabilistic no-key protocol in the encryption process of which it used the single-use shared key. Then, the no-key DE protocol has been constructed as pseudo-probabilistic no-key encryption protocol.

The proposed method and protocol for no-key DE provide resistance to passive coercive attacks. In cases when it is required to provide resistance to potential attacks performed by active coercer that impersonates the sender or receiver of secret message, one should imbed in the proposed protocol mechanism for verifying the authenticity of the data sent via communication channel. For example, one can imbed steps for authentication of the single-use public keys. The authentication mechanism can be implemented with using short (having size 16 to 56 bits) pre-shared secret keys, like in the protocol described in [5]; however, in this case, one will have a shared-key DE protocol.

The proposed no-key DE protocol provides sub-exponential deniability. To get the exponential deniability to passive coercive attacks, one can implement the proposed method using computations on elliptic curves; however, detailed consideration of this item represents a topic of an individual work.

References

1. Canetti, R., Dwork, C., Naor, M., Ostrovsky, R.: Deniable Encryption. Proceedings Advances in Cryptology-CRYPTO 1997. Lectute Notes in Computer Science. Springer-Verlag. Berlin, Heidelberg, New York, (1997), vol. 1294, 90–104
2. Meng, B.: A Secure Internet Voting Protocol Based on Non-interactive Deniable Authentication Protocol and Proof Protocol that Two Ciphertexts are Encryption of the Same Plaintext. Journal of Networks. (2009), vol. 4, no. 5, 370–377
3. Ishai, Y., Kushilevits, E., Ostrovsky, R.: Efficient Non-interactive Secure Computation. Advances in Cryptology - EUROCRYPT 2011. Lecture Notes in Computer Science. Springer-Verlag. Berlin, Heidelberg, New York. (2011), vol. 6632, 406–425
4. Wang C., Wang, J.A.: Shared-key and Receiver-deniable Encryption Scheme over Lattice. Journal of Computational Information Systems. (2012), vol. 8, no. 2, 747–753
5. Moldovyan, N.A., Moldovyan, A.A., Shcherbacov, A.V.: Deniable-encryption protocol using commutative transformation. Workshop on Foundations of Informatics. (2016) 285–298
6. Hellman M.E., Pohlig, S.C.: Exponentiation Cryptographic Apparatus and Method. U.S. Patent No 4, 424, 414. (1984)
7. Moldovyan, N.A., Berezin, A.N., Kornienko, A.A., Moldovyan, A.A.: Bi-deniable Public-Encryption Protocols Based on Standard PKI. Proceedings of the 18th FRUCT and ISPIT Conference, Technopark of ITMO University, Saint-Petersburg, Russia. FRUCT Oy, Finland. (2016) 212–219
8. Moldovyan, A.A., Moldovyan, N.A., Shcherbakov, V.A.: Bi-Deniable Public-Key Encryption Protocol Secure Against Active Coercive Adversary. Buletinul Academiei de Stiinte a Republicii Moldova. Mathematica. (2014), no. 3, 23–29
9. Diffie W., Hellman, M.E.: New Directions in Cryptography. IEEE Transactions on Information Theory. (1976), vol. IT-22, 644–654
10. Menezes, A.J., Oorschot, P.C., Vanstone, S.A.: Applied cryptography. CRC Press, New York, London, (1996)
11. Ibrahim, M.H.: A method for obtaining deniable Public-Key Encryption. International Journal of Network Security. (2009), vol. 8, no. 1, 1–9
12. Barakat, M.T.: A New Sender-Side Public-Key Deniable Encryption Scheme with Fast Decryption. KSII Transactions on Internet and Information Systems. (2014), vol. 8, no. 9, 3231–3249
13. Dachman-Soled, D.: On minimal assumptions for sender-deniable public key encryption. Public-Key CryptographyPKC 2014: 17th International Conference on Practice and Theory in Public-Key Cryptography. Lecture Notes in Computer Science. SpringerVerlag. Berlin, Heidelberg, New York. (2014), vol. 8383, 574–591