Strengthening the password authenticated key exchange protocols due to the use of asymmetric execution of cryptosystems

Varfolomeev A.A
Department of «Information Security» of
Bauman Moscow State Technical University (BMSTU)
Moscow, Russia
a.varfolomeev@mail.ru

***Abstract*— Protocols for generating high-entropy cryptographic session keys from long-term low-entropy keys or passwords (PAKE - Password Authenticated Key Exchange) are used to increase the security of information protection in information systems. The paper shows how to increase the security of the PAKE protocols themselves by using the concept of asymmetric execution of cryptosystems proposed by the author (see, for example, at SIBCON 2016, RusCrypto 2018).**

Asymmetric execution of cryptosystems implies a deliberate significant increase in the complexity of performing operations for one of the legitimate users (for example, for the server, due to additional transformations of the plaintext, increasing the password by a random value, choosing the mode of the cipher, hiding part of the information necessary for the operation), resulting in significant increasing the complexity of finding the secret keys and for the attacker.

The proposed strengthening of cryptographic protocols is especially relevant in the context of restrictions on the parameters used in the primitives of cryptographic protocol.

Keywords*— *cryptography, cryptographic protocols, keys, complexity, resilience, EKE protocol, Diffie-Hellman key agreement protocol, SESPAKE protocol.

I. INTRODUCTION

This work, in addition to independent significance, is a continuation of a number of works by the author in the field of strengthening the security of cryptosystems with short keys. Short keys are understood as keys of symmetric and asymmetric cryptosystems defined by the restriction for unlicensed use introduced by the well-known Decree of the Government of the Russian Federation dated 04.16.2012 No. 313. According to the Decree, without a license you can use a "symmetric cryptographic algorithm that uses a cryptographic key with a length not exceeding 56 bits, either an asymmetric cryptographic algorithm based either on the method of factoring integers whose size does not exceed 512 bits, or on the method of calculating a discrete logarithms in a finite field multiplicative group size not exceeding 512 bits, or a method of computing discrete logarithms in another group of size not exceeding 112 bits.. "

The concept of low-entropy keys is often used in the literature, compared with the 256-bit high-entropy keys, which are typical for modern standard encryption algorithms. Short keys can be classified as low-entropy, but a number of authors attribute password words and even PIN-codes, whose entropy is less than 56 bits, to low-entropy keys.

In [1], for symmetric ciphers, it was proposed to introduce asymmetry into the complexity of the work when encrypting plaintext and when decrypting a ciphertext by legitimate users. In this regard, such ciphers and cryptosystems were called asymmetrically executed. The legitimate recipient of the ciphertext in this case spent significantly more time decrypting, since he had to try out a random binary vector by which the short key used in encryption was increased, which was known to the sender and recipient of the message. To find the plaintext, the attacker had to try both the short key and the random vector added by the sender to the key. Of course, at the same time, it is assumed that the sender has a high-quality generator of random binary sequences, and for the model of the attacker, it is possible to use only the full testing method (brute force attack) for decryption.

To increase security, other methods were proposed in [2], including preliminary conversion of plaintext such as AON conversion. The use of new (in comparison with GOST 28147-89) operating modes of block ciphers according to the GOST 34.13 standard turned out to be essential for the implementation efficiency.

Further, at the RusCrypto 2018 conference [3], the author proposed to use the PAKE protocols to strengthen the strength of ciphers with short keys, where a short key should be used as a password word. For example, when using symmetric and asymmetric ciphers with short keys in the EKE protocol [4], the complexity of decryption is not the sum of the complexity of decryption of each of the ciphers, but is their product.

This paper shows how the introduction of asymmetry in the complexity of operations performed by legitimate users of the PAKE protocols themselves leads to an increase in the complexity of solving cryptanalysis tasks for an attacker using well-known protocols as an example.

The concept of “asymmetric PAKE” was previously considered in a number of works (see, for example, [5], [6]), but the reason for using the word “asymmetry” differs from that described in this paper. Asymmetric PAKE is the same PACK in the client-server architecture, where the server does not store the password word itself, but the value of the one way function from it. If the server is compromised, the attacker will have to perform an additional off-line attack to find the password word.

II. The Bellowin - Merritt cryptographic protocol EKE and proposals for enhancing its strength.

The EKE protocol [4] is one of the first password-based authentication key exchange protocols. The purpose of the protocol is to safely transport a high-entropy key from one user to another if they have a common low-entropy secret (password). The EKE protocol has been patented. For readability, we recall the steps and steps of the EKE protocol. Let PW denote the password word known to users A and B.

Key transportation stage.

• 1. A: sends the message [A, E (PW; PKa)] → B,
where E (PW; PKa) is the encryption transformation with a symmetric cipher on the PW key, PKa is the public key of user A for the asymmetric cipher.

• 2. B: produces a high-entropy key k and sends
E (PW; E_PKa (k)) → A.
The step of confirming receipt of key k is “Request-response”.

• 3. A: From E (PW; E_PKa (k)) receives k, sends
E (k; R_a) → B.

• 4. B: From E (k; R_a) it receives R_a, sends
E (k; h (R_a) II R_b) →A, here h is some hash function.

• 5. A: From E (k; h (R_a) II R_b) receives h (R_a) II R_b and checks h (R_a) = ?, then sends E (k; h (R_b)) → B.

• 6. B: From E (k; h (R_b)) receives and checks h (R_b) =?.

According to the Dolev - Yao model, the attacker in step 1 receives ciphertext E (PW; PKa) for off-line attacks. Due to the relatively small number of options for choosing PW, he can go through all of them and get options for the PKa public key. But this key, which is clear text for E (PW; PKa), has no structure and is similar to a random sequence of bits. Therefore, he cannot isolate the true password word at this step.

In step 2, each selected PW option leads to the task of finding the key k by the corresponding public key PKa from step 1 and the ciphertext E_PKa (k) obtained by decrypting E (PW; E_PKa (k)) to PW. Even if it is possible to test all the secret keys of the asymmetric encryption algorithm, it is impossible to determine the true key k, since it also has no structure. With short public and secret keys of the asymmetric algorithm (for example, 512 bits), a faster finding of k is possible, but still among the options for PW.

It follows that, to increase the complexity of finding the key k by an attacker, it is necessary to increase the number of options for PW.

Here are options for enhancing the strength of the EKE protocol only for the key transportation phase, omitting the second stage for brevity.

1 option for protocol amplification.

• 1. A: [A, E (PW-Ra; PKa II h (PKa)] → B, where Ra is a random vector selected by user A and not known to user B.

• 2. B iterates over Ra and searches for Pka by the criterion for plaintext. Then it sends E (PW; E_PKa (k)) → A.

In step 1, a hash code from it was added to the plaintext PKa in order to structure this plaintext so that user B could find the true public key PKa of user A. Decryption time for B was increased by 2 ^ IRaI, where IRaI is the size vectors Ra.

But the criterion for plaintext also gets the attacker. Thus, this case is similar to the case of transmitting ciphertext obtained by encrypting structural plaintext on the PW key (text structure of the form PKa II h (PKa) is chosen for clarity). To enhance the strength of encryption in this case, the recommendations of previous works [1-2] can be applied.

Option 2 protocol amplification.

At the first step, a certain set of public keys of user A is encrypted: PKa1, PKa2, ..., PKaN.

• 1. A: [A, E (PW; PKa1, PKa2, ..., PKaN)] → B.

• 2. B: After decrypting the ciphertext E (PW; PKa1, PKa2, ..., PKaN) and receiving the set of public keys {PKa1, PKa2, ..., PKaN}, user B randomly chooses one of them - the PKaJ key.

• Next, B generates a high-entropy key k and sends message A to user A (PW; E_PKaJ (k II h (k)) → A.

• 3. A, knowing the password word PW, decrypts E (PW; E_PKaJ (k II h (k)) and restores E_PKaJ (k II h (k), iterates over its public keys PKa1, PKa2, ..., PKaN, looks for PKaJ and key k. The criterion for finding the true key k is finding the key k with the correct value h (k) attached.

User A in this option works more than B. In step 3, he must sort through all N sent to the user B public keys PKa1, PKa2, ..., PKaN.

In step 1, as before, the sequence PKa1, PKa2, ..., PKaN is not structured, which does not allow an attacker to recover the password word PW from the ciphertext E (PW; PKa1, PKa2, ..., PKaN)] even with full testing.

At step 2, we have E_PKaJ (k II h (k)) - the ciphertext obtained by encrypting the text (k II h (k)) on the public key PKaJ of user A. It has no structure, which does not allow the attacker to find PW from the ciphertext E (PW ; E_PKaJ (k II h (k)) even by full testing.

After step 1, an attacker can have IPWI variants of the set of public keys {PKa1, PKa2, ..., PKaN}, where IPWI is the number of possible variants of the word PW.

After 2 steps, the attacker has IPWI * N decryption tasks for the ciphertext E_PKaJ (k II h (k). To assess the complexity of decryption, one must proceed from the strength of the asymmetric algorithm used.

3 option for protocol amplification.

• 1. A: [A, E (PW - Ra; PKa1, PKa2, ..., PKaN, h (*))] → B.

• 2. B: Iterates over Ra and looks for a true set of public keys. The criterion is the structure of the text [PKa1, PKa2, ..., PKaN, h (*)]. Selects one PKaJ key of user A.

• E (PW-Ra; E_PKaJ (k II h (k)) → A, k is a high-entropy key.

• 3. A: iterates over its public keys PKa1, PKa2, ..., PKaN (Ra - knows), searches for PKaJ and key k. The criterion is finding the key k with the correct value h (k) attached.

4 option protocol amplification.

• 1. A: [A, E (PW - Ra; PKa1, PKa2, ..., PKaN, h (*))] → B.
• 2. B: Iterates over Ra and looks for a true set of public keys. The criterion is the structure of the text [PKa1, PKa2, ..., PKaN, h (*)]. Selects one PKaJ key of user A.
• E (PW-Rb; E_PKaJ (k II h (k)) → A, k is a high-entropy key.
• 3. A: iterates over its public keys PKa1, PKa2, ..., PKaN and Rb, searches for PKaJ and key k. The criterion is finding the key k with the correct value h (k) attached.

In the fourth embodiment, the complexity of the user A increases due to the unknown vector Rb and the choice of one of N public keys by the user B, and the complexity of the user B increases due to the unknown vector Ra.

In option 3, the vector is Rb = Ra and is known to user A.

But in all these cases, the complexity of the decryption task for the attacker also increases.

Diffie-Hellman cryptographic key agreement protocol and suggestions for enhancing its strength.

Another type of key installation protocols, in addition to key transportation protocols, are key agreement protocols [7-8]. In the key agreement protocol, none of the participants knows in advance which key will be installed for communication with other participants. This key will be generated as a result of the exchange of some information between the participants in the interaction. The most famous example of a key agreement protocol is the Diffie and Hellman protocol [9].

Recall it for the case of using a group of points of elliptic curves.

A, B - participants in the protocol.

P is the base point of the elliptic curve. 112 bits is a security setting.

Xa is the secret key of participant A, a natural number.

Ya = Xa * P - public key of A → B.

Xb is the secret key of participant B, a natural number.

Yb = Xb * P - public key of B → A.

A: calculates the shared key K = Xa * Yb = (Xa * Xb) * P

B: computes the shared key K = Xb * Ya = (Xb * Xa) * P

It is known that for the safe use of this protocol, it is necessary to add an authentication channel to it in order to exclude a "man in the middle" attack. In particular, the password word is also used for this. There are many ways to do this.

But first, we will demonstrate how to strengthen this protocol using asymmetric operations, but to modify the Diffie-Hellman protocol to the transport protocol.

Gain option.

Participant A sends a set of {Ya1, ..., YaL} public keys to Participant B.

Participant B sends a set of {Yb1, ..., YbL} public keys to Participant A.

For clarity, the power sets of public keys are selected the same. In general, these capacities may be different.

Participant A randomly selects the public key YbJ and his private key XaS, builds a key of the form K = XaS * YbJ.

Then he can use this key, for example, to encrypt some plaintext OT on the key K and send it to participant B:

C = E (K, OT) → B.

Participant B, in order to find K and decrypt C, must sort through L variants of his secret and L variants of Participant A public keys. The criterion for the correct choice is the criterion for clear text. The complexity for B is in order equal to the number L ^ 2 times the complexity of the operation of decryption and application of the plain text criterion. For an attacker who also has all the public keys of the participants, the task is to solve L ^ 2 discrete logarithm problems (albeit with a parameter of 112 bits).

In the proposed variant of protocol amplification, we lose in the so-called communication complexity of the protocol when transmitting sets of public keys L ^ 2 times. The complexity of the decryption process for participant B also increases by the same amount. The security parameter is 112 bits, selected by license restriction, it suggests that plaintext and key can in principle be found, but at very high computational costs. The choice of the parameter L may complicate this possibility.

SESPAKE cryptographic protocol and suggestions for enhancing its strength.

An authentication channel for the Diffie-Hellman protocol can be provided using digital signatures or a shared secret (password). Many options are suggested for this. Consider the SESPAKE protocol, where participants are supposed to have a common secret word PW.

Recall the main steps of this protocol. Details can be found in [6]. This protocol also uses a group of points of an elliptic curve (In our consideration, the security parameter is 112 bits).

A is the client, B is the server. Both store L points {Q_1, ..., Q_L} and point P of the elliptic curve. Participant B additionally stores:

number ind from the set {1, ..., L}, string salt, point Q_pw = int (f (PW, salt)) * Q_ind. Password words PW may not be stored.

1. B → A: ind, salt.

2. A computes Qa_pw = int (f (PW, salt) * Q_ind. (=

Q_pw of server B).

A → B: Ya = Xa * P - Qa_pw. (= Xa * P - Q_pw)

3. B calculates Qb = Ya + Q_pw, (= (Xa * P - Q_pw) + Q_pw = Xa * P)

B →A: Yb = Xb * P + Q_pw.

4. A: Qa = Yb - Qa_pw = Xb * P + Q_pw - Qa_pw = Xb * P.
Xa * Xb * P gets → Ka,

B: Qb = Ya + Q_pw = (Xa * P - Qa_pw) + Q_pw = Xa * P.
Xb * Xa * P gets → Kb.

Ka = Kb.

(Here it is omitted how specifically Ka and Kb are obtained, it is enough that the relation Xa * Xb * P = Xb * Xa * P is fulfilled).

It can be seen from the above description that the SESPAKE protocol is a variant of the Diffie-Hellman protocol with public keys, distorted by the function of the password word PW, known only to legitimate participants.

Option with asymmetric execution.

The basic idea is not to pass the number ind. Participant A randomly selects ind from the set {1, ..., L} and computes Ya, but participant B sends L public keys.

A is the client, B is the server. Both store L points {Q_1, ..., Q_L} and point P of the elliptic curve. Participant B additionally stores: string salt, points Q_pw1 = int (f (PW, salt)) * Q_1, ...,. Q_pwL = int (f (PW, salt)) * Q_L. The password word PW may not be stored.

1. B → A: salt.

2. A computes for random Qa_pwJ = int (f (PW, salt)) * Q_J.

A → B: YaJ = Xa * P - Qa_pwJ.

3. B → A: Yb1 = Xb + Q_pw1, ..., YbL = Xb + Q_pwL.

A: chose YbJ and Qa = YbJ - Qa_pwJ = (Xb * P + Q_pwJ) - Qa_pwJ = Xb * P.

B calculates Qb1 = YaJ + Q_pw1, ..., QbL = YaJ + Q_pwL. Among them, with the right Q_pwJ.

QbJ = YaJ + Q_pwJ = (Xa * P - Qa_pwJ) + Q_pwJ = Xa * P.

4. A: The Ka key is used to encrypt plaintext OT.

This work does not provide specific parameters for enhancing durability, since in many respects they depend on the computing power of the attacker and the computing power of the legitimate participants in the interaction, as well as on the requirement for the speed of obtaining information, which in turn can be dictated by the necessary level of security in each specific case.

III. CONCLUSIONS

The technology of asymmetric execution of cryptosystems in the PAKE protocol can increase the security of these protocols. In this paper, this is demonstrated on three protocols. For each of them, specific methods for asymmetric execution are proposed.

REFERENCES

[1] Varfolomeev A.A. Nekotorye rekomendacii po povysheniyu stojkosti shifra s malym razmerom klyucha k metodu polnogo oprobovaniya. Voprosy kiberbezopasnosti [Cybersecurity issues], 2015, No 5(13), pp. 60-62.

[2] Varfolomeev A.A. O nekotoryh predvaritel'nyh preobrazovaniyah otkrytogo teksta tipa «All-Or-Nothing» dlya usileniya stojkosti shifra k metodu polnogo oprobovaniya. SIBCON 2016. http://ni.spbstu.ru/wp-content/uploads/2016/05/SibCon-2016-programa.pdf .

[3] Varfolomeev A.A. Ob asimmetrichno vypolnimyh simmetrichnyh kriptosistemah (shifrah). RusKripto 2018.

[4] Bellovin, S. M.; M. Merritt (May 1992). Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks. Proceedings of the I.E.E.E. Symposium on Research in Security and Privacy. Oakland. p. 72. doi:10.1109/RISP.1992.213269. ISBN 978-0-8186-2825-2.

[5] Jareck, S., Krawczyk H., Xu J. (2018). OPAQUE: An Asymmetric PAKE Protocol Secure Against Pre-Computation Attacks. *Advances in Cryptology. Lecture Notes in Computer Science. 10822. pp. 456–486.* doi:10.1007/978-3-319-78372-7_15. ISBN 978-3-319-78371-0.

[6] RFC 8133. The Security Evaluated Standardized Password-Authenticated Key Exchange (SESPAKE) Protocol. *https://tools.ietf.org/html/rfc8133*

[7] Slovar' kriptograficheskih terminov. / Pod red. B. A. Pogorelova i V. N. Sachkova. - M.: МЦНМО, 2006. - 94 с.

[8] Menezes A., van Oorschot P., Vanstone S., (1997). Handbook of Applied Cryptography, Boca Raton, CRC Press. ISBN 0-8493-8523-7. (Available online)

[9] Diffie W., Hellman M., New Directions in Cryptography, IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, pp: 644—654.

[10] Smyshlyaev S., Oshkin I., Alekseev E., Ahmetzyanova L. (2015). "On the Security of One Password Authenticated Key Exchange Protocol". Cryptology ePrint Archive (Report 2015/1237).