UDC: 007
Original scientific paper

A SECURE KEY AGREEMENT PROTOCOL

Constantin Popescu

University of Oradea, Department of Mathematics, Oradea, Romania
E-mail: cpopescu@math.uoradea.ro

In this paper we propose a secure protocol for an authenticated key agreement based on the Diffie-Hellman key agreement, which works in an elliptic curve group We prove that our protocol meets the security attributes under the assumption that the elliptic curve discrete logarithm problem is secure.

Keywords: authenticated key agreement, protocol, Diffie-Hellman, elliptic curve.

1. INTRODUCTION

In a key agreement protocol two or more distributed entities need to share some key in secret, and this is called a session key. Numerous Diffie-Hellman-based key agreement protocols have been proposed over the years [1], [5], [11], [12]. But many of them have turned out to be flawed [2], [10]. A number of desirable attributes of the key agreement protocols have also bean identified [19] and nowadays most protocols are analyzed with such attributes.

The authors [8] proposed a new authenticated key agreement protocol, which is resistant to a small subgroup attack and to an unknown key-share attack and has some computational advantage with about 2.5 integer multiplications for each entity. However, Kaliski showed in [5] that this protocoldoes not possess tine unknown key-share attribute.

In this paper, we propose a secure key agreement protocol. The protocol is based on the Diffie-Hellman key agreement [3], and has the desirable attributes discussed in [19]. We will also present a multiple key agreement protocol which enables the participants to share two or more keys in one execution of the protocol. The protocols described in this paper have been described in the setting of the group of points on an elliptic curve defined over a finite field. Suitable choices include the multiplicative group of a finite field, subgroups Z^*_n, where n is a composite integer, and subgroups of Z^*_q of prime order q. Elliptic curve groups are advantageous because they offer the same security as other groups but with smaller key sizes and faster computation times.

2. A DESCRIPTION OF THE KEY AGREEMENT PROTOCOL

Many researchers have examined elliptic curve cryptosystems, which were firstly proposed by Miller [13] and Koblitz [6]. The elliptic curve cryptosystems, which are based on the elliptic curve logarithm over a finite field, have some advantages over other systems: the key size can be much smaller than the other schemes since only exponential-tune attacks have been known so far, if the curve is carefully chosen [7], and the elliptic curve discrete log-

arithms might be still intractable even if factoring and the multiplicative group discrete logarithm are broken.

In this section we will describe the proposed key agreement protocol which is specified by the key generation and the protocol description.

2.1. Key Generation

In order to avoid the Pollard-rho [16] and Pohling-Hellman [15] algorithms for the elliptic curve discrete logarithm problem, it is necessary that the number of F_q-rational points on E, denoted $\#E(F_q)$, be divisible by a sufficiently large prime n. To avoid the reduction algorithms of Menezes, Okamoto and Vanstone [9] and Frey and Ruck [4], the curve should be non-supersingular (i.e., p should not divide $(q+1-\#E(F_q))$). To avoid the attack of Samaev [17], Smart [18] on F_q- anomalous curves, the curve should not be F_q-anomalous (i.e., $\#E(F_q) \neq q$).

Firstly, we will choose the elliptic curve domain parameters:

1. a field size q, where q is a prime power (in practice, either $q=y$, an odd prime, or $q=2^m$).
2. two field elements a, $b \in F_q$, which define the equation of the elliptic curve E over F_q (i.e., $y^2 = x^3 + ax + b$ in the case $p > 3$), where $4a^3 + 27b^2 \neq 0$.
3. two field elements x_p and y_p in F_q, which define a finite point $P = (x_p, x_y)$ of prime order in $E(F_q)$ ($P \neq O$, where O denotes the point at infinity).
4. the order n of the point P.

The operation of the key generation is as follows:

1. Selects P of order n in the group $E(F_q)$.
2. Let H be a one-way hash function such as SHA-1 [14].
3. Selects random integers s_A, s_B from the interval [1, n- 1]. The value s_A is a secret key of the user A and s_B is the secret key of the user B.
4. Computes the points $Y_A = -s_A \cdot P$ and $Y_B = -s_B \cdot P$, which are the public key of a user A and B respectively.
5. Let ID_A be an identity information of a user A and ID_B, can be the identity information of a user B.

2.2 Protocol Description

The key agreement protocol between A and B is as follows:

1. A generates random integers r_A, k_A (ephemeral keys) from the interval [1, n – 1] and computes Q_A, V_A, points on E, such that

$$Q_A = r_A \cdot P,\ V_A = -k_A \cdot P.$$

A sends the point V_A to B.

2. B randomly selects integers r_B, k_B (ephemeral keys) from the interval [1, n - 1] and computes Q_B, V_B, points on E, such that

$$Q_B = r_B \cdot P,\ V_B = -k_B \cdot P.$$

B computes $e_B = H(x_{Q_B}, x_{V_B}, x_{V_A}, ID_B, ID_A)$ and $d_B = r_B + e_B k_B + e_B s_B$, where x_{Q_B}, is

the x-coordinate of Q_B, x_{V_A} is the x-coordinate of V_A and x_{V_B} is the x-coordinate of V_B. B sends V_B, e_B, d_B to A.

3. A computes the point U_B such that. $U_B = d_B \,.\, P + e_B \,.\, V_B + e_B \,.\, Y_B$ and checks if $e_B = H(x_{U_B}, x_{V_B}, x_{V_A}, ID_B, ID_A)$. If it does not hold, then A terminates the execution. Otherwise, A computes

$$e_A = H(x_{Q_A}, x_{V_A}, x_{V_B}, ID_A, ID_B)$$
$$d_A = r_A + e_A k_A + e_A s_A$$

where x_{U_B} is the x-coordinate of U_B, x_{V_B} is the x-coordinate of V_B, x_{Q_A} is the x-coordinate of Q_A and x_{V_A} is the x-coordinate of V_A. A computes the point K_A, such that

$$K_A = -k_A \,.\, V_B.$$

and sends e_A, d_A to B.

4. B computes the point U_A, such that $U_A = d_A \,.\, P + e_A \,.\, V_A + e_A \cdot Y_A$ and checks if $e_A = H(x_{U_A}, x_{V_A}, x_{V_B}, ID_A, ID_B)$. If it does not hold, then B terminates the execution. Otherwise, B computes

$$K_B = -k_B \,.\, V_A.$$

The shared secret is the point $K = K_A = K_B$.

3. THE MULTIPLE KEY AGREEMENT PROTOCOL

In this section we will present a multiple key agreement, protocol which enables the participants to share two or more keys in one execution of the protocol. The key generation is the same as in Section 2. The multiple key agreement protocol between A and B is as follows:

1. A generates random integers r_A, k_{A_1}, ..., k_{A_n} from the interval [1, n – 1] and computes the points Q_A, V_{A_i}, $i = 1,..., n$, such that

$$Q_A = r_A \cdot P,\ V_{A_i} = -k_{A_i} \cdot P.$$

A sends the points V_{A_i} $i = 1, ..., n$ to B.

2. B randomly selects integers r_B, k_{B_1}, ..., k_{B_n} from the interval [1, n – 1] and computes Q_B, V_{B_i} $i = 1, ..., n$ such that

$$Q_B = r_B \cdot P,\ V_{B_i} = -k_{B_i} \,.\, P.$$

B computes

$$e_B = H(x_{Q_B}, x_{V_{B_1}}, ..., x_{V_{B_n}}, x_{V_{A_1}}, ..., x_{V_{A_n}}, ID_B, ID_A)$$

$$d_B = r_B + e_B \sum_{i=1}^{n} k_{B_i} + e_B s_B$$

where x_{Q_B} is the x-coordinate of Q_B, $x_{V_{B_i}}$ is the x-coordinate of V_{B_i} and $x_{V_{A_i}}$ is the x-coordinate of V_{A_i} $i = 1, ..., n$. B sends V_{B_i}, $i = 1, ..., n$, e_B, d_B to A.

3. A cumputes the point U_B, such that $U_B = d_B \cdot P + e_B \sum_{i=1}^{n} \cdot V_{B_i} + e_B \cdot Y_B$ and checks if $e_B = H(x_{U_B}, x_{V_{B_1}}, ..., x_{V_{B_n}}, x_{V_{A_1}}, ..., x_{V_{A_n}}, ID_B, ID_A)$. If it does not hold, then A terminates the execution. Otherwise, A computes

$$e_A = H(x_{Q_A}, x_{V_{A_1}}, ..., x_{V_{A_n}}, x_{V_{B_1}}, ..., x_{V_{B_n}}, ID_A, ID_B)$$

$$d_A = r_A + e_A \sum_{i=1}^{n} k_{A_i} + e_A s_A$$

where x_{Q_A} is the x-coordinate of Q_A, $x_{V_{A_i}}$, is the x-coordinate of V_{A_i}, x_{V_B} is the x-coordinate of V_{B_i}, $i = 1, ..., n$. A computes the points K_{A_i}, such that

$$K_{A_i} = -k_{A_i} \cdot V_{B_i}, \quad i = 1, ..., n.$$

and sends e_A, d_A to B.

4. B computes the point U_A, such that $U_A = d_A \cdot P + e_A \sum_{i=1}^{n} V_{A_i} + e_A \cdot Y_A$ and checks if $e_A = H(x_{U_A}, x_{V_{A_1}}, ..., x_{V_{A_n}}, x_{V_{B_1}}, ..., x_{V_{B_n}}, ID_A, ID_B)$, where x_{U_A} is the x-coordinate of U_A, x_{V_A}, is the x-coordinate of V_{A_i}, $x_{V_{B_i}}$, is the x-coordinate of V_{B_i}, $i = 1, ..., n$. If it does not hold, then B terminates the execution. Otherwise, B computes

$$K_{B_i} = -k_{B_i} \cdot V_{A_i}, \quad i = 1, ..., n.$$

The shared secret keys are the points $K_i = K_{A_i} = K_{B_i}$, $i = 1, ..., n$.

4. SECURITY CONSIDERATIONS

We will prove that our protocol meets the following desirable attributes under the assumption that the elliptic curve discrete logarithm problem is secure.

Known-Key Security: If the two entities A and B execute the regular protocol run, then they clearly share their unique session key K as above.

(Perfect) Forward Secrecy: During the computation of the session key K for each entities, the random integers r_A, k_A, r_B, k_B still act on it. An adversary who captured their private keys s_A or s_B should extract the random integers (ephemeral key) r_A, k_A, r_B, k_B from the information Q_A, V_A, Q_B, V_B to know the previous or next session key between them. But, this is the elliptic curve discrete logarithm problem.

Key-compromise Impersonation: Now, suppose the long-term private key s_A of the user A is disclosed. An adversary who knows this value can clearly impersonate A. Also, the adversary impersonates B to A knowing B's long-term private key s_B. For the personation to succeed the adversary must know A's ephemeral keys r_A and k_A. Also, in this case, the adversary should extract r_A and k_A from A's ephemeral public value, Q_A, and V_A, to generate the same session key K with A. This also is the elliptic curve discrete logarithm problem.

Key Control: The key-control is impossible for a third party. The only possibility of a key-control attack may be brought on by the participation of the protocol B. But for the party

B to make the party A generate the session key K_B which is a preselected by B, for example B should solve the equation $K_B = -k_B \cdot V_A$. This is the elliptic curve discrete logarithm problem.

Unknown Key-Share: Suppose an adversary C tries to make A believe that the session key is shared with B, while B believes that the session key is shared with C. To launch the unknown key-share attack, the adversary C should set its public key to be certified even though he does not know its correct private key. For this C makes it by utilizing the public values (points) Y_A, Y_B and P. Let $f_t(R_1,..., R_l) = \sum_{i=1}^{l} t_i R_i$, where R_i's are points on E and $t = (t_1,...,t_l)$ are integers from the interval [1, n - 1]. Then C should set his public key Yc as $Yc = f_t(Y_A,, Y_B, P)$. Suppose C got the value Yc certified as its public key and let's suppose the following generalized model for an unknown key-share attack: Suppose that $Vc = f_p(Y_A, Y_B, P, V_B)$ and V_C's $= f_m(Y_A, Y_B, P, V_A)$, where $p = (p_1, ..., p_l)$ and $m = (m_1, ..., m_l)$ are integers from the interval [1, n - 1]. For C to launch the unknown key-share attack successfully, it should force A and B to share the same secret session key $K = K_A = K_B$ through the protocol run. In practice, through the protocol run, A and B get their session key K_A and K_B respectively as in the case with those keys in the following equation:

$$K_A = -k_A \cdot V_B,\ K_B = -k_B \cdot V_C'.$$

The adversary C does not know s_A, s_B, k_A, k_B even though C can control the integer values t_i, p_i, m_i. The adversary C can force the equation $K_A = K_B$ to hold for many values of k_A and k_B. Now we can consider the following equation as an identical one for the variables k_A and k_B

$$k_A \cdot V_B = k_B \cdot V_C'.$$

We can change this equation as the form $a \cdot P = O$, by unfolding the values V_A, Y_C, V_C, V_C' with respect to P. Then we are unable to solve equation t_i, p_i, m_i, since we do not have sufficient information on s_A, s_B, k_A, k_B.

5. CONCLUSION

In this paper we proposed a secure protocol for authenticated key agreement based on the Diffie-Hellman key agreement, which works in an elliptic curve group.

One disadvantage is that each participant has to generate two random numbers from the interval [1, n - 1] in one execution. Another disadvantage is that it requires entities slightly more modular exponentiations or integer multiplications than other protocols (e.g. the protocol of Law, Menezes, Qu, Solinasand Vanstone [8] or the protocol of Blake-Wilson, Johnson and Menezes [19]). But, we have proven that our protocol meets the security attributes (inclusive of the attribute of the unknown key-share) under the assumption that the elliptic curve discrete logarithm problem is secure.

REFERENCES

[1] Bellare, M., P. Rogaway (1994). Entity Authentication and Key Distribution. *In Advances in Cryptalogy CRYPTO'93*, pp. 341-358.

[2] Burmester, M. (1994). On the risk of opening distributed keys. *In Advances in Cryptology CRYPTO'94*, pp. 308-317.

[3] Diffie, W., M. Hellman (1976). New directions in cryptography. *IEEE Trans. Inform. Theory, IT 22,* No. 6, pp. 644-654.

[4] Frey, G., H. Ruck (1998). A remark concerning *m*-divisibility and the discrete logarithm in the divisor class group of curves. *Mathematics of Computation,* Vol. 67, pp. 353-356.

[5] Kaliski, B. (1998). Contribution to ANSI X9F1 and IEEE P1363 working groups.

[6] Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of Computation,* Vol. 98, pp. 203-209.

[7] Koblitz, N (1992). CM-Curves with Good Cryptographic Properties. *Proceedings of Crypto'91.*

[8] Law, L., A. Menezes, M. Qu, J. Solinas, S. Vanstone (1998). An efficient Protocol for Authenticated Key Agreement. *Technical Report CORR98-05,* Department of CO, University of Waterloo.

[9] Menezes, A., T. Okamoto, S. Vanstone (1993). Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory,* Vol. 39, pp. 1639-1646.

[10] Menezes, A., P. van Oorschot, S. Vanstone (1997). *Handbook of Applied Cryptography.* CRC press.

[11] Menezes, A., M. Qu, S. Vanstone (1995). Key Agreement and the need for authentication. PKS'95, Toronto, Canada.

[12] Menezes, A., M. Qn, S. Vanstone (1995). Some new key agreement protocols providing mutual implicit authentication. *Workshop on Selected Areas in Cryptography (SAC'95),* pp. 22-32.

[13] Miller, V. (1986). Uses of elliptic curves in cryptography. In Advances in Cryptology, Proceedings of Crypto'85, *Lecture Notes in Computer Sciences,* Springer-Vcrlag, pp. 417-426.

[14] National Institute of Standards and Technology (1995). Secure Hash Standard (SHS). FIPS Publication 180-1, April 1995.

[15] Pohling, S., M. Hellman (1978). An improved algorithm for computing logarithms over *GF(p)* and its cryptographic significance. *IEEE Transactions on Information Theory,* Vol. 24, pp. 106-110.

[16] Pollard, J. (1978). Monte Carlo methods for index computation *mod p. Mathematics of Computation* Vol. 32, pp. 918-924.

[17] Semaev, I. (1998). Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. *Mathematics of Computation,* Vol. 67, pp. 353-356.

[18] Smart, N. (1997). The discrete logarithms problem on elliptic curves of trace one. Preprint.

[19] Wilson, S., D. Johnson, A. Menezes (1997). Key Agreement Protocols and Their Security Analysis. Sixth IMA International Conference on Cryptography and Coding, Cirencester, England.

Received: 17 April 2000
Accepted: 15 October 2001

Constantin Popescu

PROTOKOL ZA USUGLAŠAVANJE SIGURNOSNIH KLJUČEVA

Sažetak

U ovom radu radi se o sigurnosnom protokolu za usuglašavanje sigurnosnih ključeva zasnovanom na Diffie-Hellmanovon ključeva. Diffie-Hellmanov sporazum ključeva radi u grupi eliptičkih krivulja. U radu se dokazuje da uvedeni protokol zadovoljava sigurnosna svojstva, pod uvjetom da je problem diskretnih logaritama eliptičkih krivulja siguran.

Ključne riječi: usuglašavanje autentifikacijskih ključeva, protokol, Diffie-Hellman, eliptičke krivulje.