Annals of the University of Craiova, Mathematics and Computer Science Series
Volume 37(2), 2010, Pages 83–91
ISSN: 1223-6934

Authentication protocol based on elliptic curve cryptography

Nicolae Constantinescu

Abstract. The authentication protocols based on classic cryptography use public-key cryptosystems for establishing the common key. Some of them have been proven to be secure but they require high amount of resources and they need large keys. Applying an elliptic curve authentication protocol the memory and the power consumption are lower. Another advantage is that this kind of protocols are secure enough even if a small key is used. In this paper we present an authentication protocol based on elliptic curve cryptography and zero knowledge. The protocol is developed for group communication. Every member of the group has a secret information and the communication starts when all this information is put together. So, if one member is not online, the others cannot communicate. We, also, present some situations where this kind of protocol is needed.

2010 Mathematics Subject Classification. Primary 94A55; Secondary 11T71, 68P25.
Key words and phrases. elliptic curve, cryptosystem, authentication protocol.

1. Introduction

Victor Miller and Neal Koblitz have, independently, proposed using elliptic curve in cryptography in 1985 [13]. In the past years, using elliptic curves in cryptography has become more and more popular. They are mainly used in coding theory, pseudorandom bit generation and number theory [12].

Definition 1.1. *An elliptic curve E over the field F is given by:* $y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$

The cryptographic elliptic curves are defined over a finite field F_p and have an easier equation:

$$y^2 = x^3 + ax + b \tag{1}$$

p is a prime number called the characteristic of F and $pa = 0\ \forall a \in F$. The above equation is given for $p \neq 2, 3$. If the discriminant $\Delta = 4a^3 + 27b^2 (mod\ p) \neq 0$ the curve E is defined as the set of points (x, y) satisfying the equation (1) including the infinity point O. The addition and the multiplication are defined below:

Definition 1.2. *Let $P(x, y)$ be a point on E. The inverse will be $-P(x, -y)$. Let $P(x_1, y_1) \in E$ and $Q(x_2, y_2) \in E$ then $P + Q = R(x_3, y_3)$ is given by:*

$$x_3 = \lambda^2 - x_1 - x_2$$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

where $\lambda = \frac{y_1 - y_2}{x_1 - x_2}$. For doubling a point $2P(x_3, y_3)$ we use the formulas:

$$x_3 = \lambda^2 - 2x_1$$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

Received April 02, 2010. Revision received May 21, 2010.

where $\lambda = \frac{3x_1^2+a}{2y_1}$.

The elliptic curves over a binary field are given by the same equation (1). The discriminant is $\Delta = b$, and for a point $P(x, y)$ the inverse is $-P(x, x + y)$. Addition and doubling are computed in the same way as on the prime curves. An elliptic curve cryptosystem performance depends the most on the scalar multiplication operation $kP = Q$ (a point $P \in E$ is multiplied by an integer k resulting a point $Q \in E$). This operation is computed through a combination of point-additions and point-doublings. In [23, 17, 10] it is recommended to use successive doubling and binary expansion. The method is efficient even if k is a large number. The points' coordinates are reduced modulo p, so the numbers involved are kept small. One of these methods is the Montgomery's Point Multiplication method presented below.
INPUT: An integer k and a point $P \in E(F_q)$
OUTPUT: $Q = kP$

Algorithm 1 Montgomery's Point Multiplication Method

```
k ← (k_{n_k−1} ... k_1 k_0)_2
P_1 ← P
P_2 ← 2P_1
for i = n_k − 2 downto 0 do
   if k_i = 1 then
      P1 ← P_1 + P_2
      P2 ← 2P_2
   else
      P2 ← P_2 + P_1
      P_1 ← 2P_1
   end if
end for
q ← P_1
return Q
```

Also, performant algorithms have been developed for computing scalar multiplication for elliptic curves with special properties. Most of them use the Frobenius map to represent k. Koblitz [14] and Muller [15], also, proposed the Frobenius map representation. Like some classic cryptosystems' security is based on the DLP (Discrete Logarithm Problem), the elliptic curve cryptosystems' security is based on the ECDLP (Elliptic Curve Discrete Logarithm Problem). This states that given $P \in E$ and $Q = kP \in E$, k is very hard to find (almost impossible). For some curves the ECDLP has been solved efficiently [20]. To avoid this problem the elliptic curve must be chosen carefully. NIST recommends fifteen elliptic curves. Specifically, FIPS 186-3 has ten recommended finite fields. There are five prime fields F_p for $p = 192, 224, 256, 384, 521$. For each of the prime fields one elliptic curve is recommended. There are five binary fields F_{2^m} for $2^{163}, 2^{223}, 2^{283}, 2^{409}, 2^{571}$. For each of the binary fields one elliptic curve and one Koblitz curve was selected. The curves were chosen for optimal security and implementation efficiency [8, 3]. The table below shows NIST's recommendation on choosing equivalent symmetric and public key sizes [16].

Symmetric	ECC	RSA/DH/DSA	MIPS Yrs to attack	Protection Lifetime
80	160	1024	10^{12}	Until 2010
112	224	2048	10^{24}	Until 2030
128	256	3072	10^{28}	Beyond 2031
192	384	7680	10^{47}	Beyond 2031
256	512	15360	10^{66}	Beyond 2031

2. Related Work

In [4, 19] the authors present an authentication protocol for exchanging encrypted messages via an authentication server based on elliptic curve cryptography using El Gamal's algorithm. They chose El Gamal's algorithm for encryption/decryption and, also, for the authentication. The authors motivated this by the fact that using two or more algorithms in the same protocol makes it more vulnerable. They, also, pointed out that using a digital signature algorithm is more secure, but they preferred not to use one because the presented protocol has a big advantage: the receiver does not need to know the senders public key [2]. A short description of the protocol is presented below.

$SendingProcess(From = Alice, To = Bob)$:
$A \rightarrow S : C_S(From, To, N_1)$
$S \rightarrow A : C_A(C_B, N_1, N_2, Id)$
$A \rightarrow S : C_B(M, N_2), Id$
$ReceivingProcess(Bob\ receives\ the\ message)$:
$B \rightarrow S : C_S(To, N_3)$
$S \rightarrow B : C_B(M, N_2), C_B(N_2, N_3)$

First, the user A sends a request to the server S consisting in the nickname of the sender (From), the nickname of the receiver (To) and a random message (N_1). This random message is used to verify the server. These elements are encrypted with the server's public key and sent to the server. After decryption, the server encrypts a mail Id, N_1, a new N_2, the receiver's (B) public key with A's public key, and sends the result to the sender (A). The sender verifies N_1 to see if the message received from the server is valid. Next, A encrypts the clear message M and N_2 with B's public key and sends the encryption to S along with the Id. To download his message, B send his nickname and a new random message N_3 encrypted with S's key. The server S sends back to B M and N_2 encrypted with B's key along with N_2 and N_3, also encrypted with B's key. To verify if the encryption received is from the server S, B checks N_3. In [1] are introduced two authentication protocols which use two different algorithms: one for encrypting the message and one for generating the key. In both cases, the encryption algorithm is a symmetric one. The authors recommend the protocol presented in the first paper to be use for controller-pilot data link communications, while the one presented in the second paper is recommended for the wireless communication, but it does not mean that they cannot be adapted for other communications. They both use the Elliptic Curve Digital Signature Algorithm (ECDSA) for authentication. This algorithm is briefly described below. In [5] it is presented an authentication rank protocol based on elliptic curve infrastructure.

ECDSA. The communicating parties choose an elliptic curve E defined over F_p or F_{2^m} with large group of order n and a point P. All this information are made public. The algorithm, like all the digital signature algorithms, has three steps:

(1) **Key Generation**
   (a) The first user selects a random integer $k_1 \in [2, n-2]$;
   (b) Then the user computes $Q = k_1 P$;
   (c) The public and the private key of the first user are (E, P, n, Q) and k_1, respectively.
(2) **Signature Generation**
   (a) The first user selects an integer $k_2 \in [2, n-2]$;
   (b) Then computes $k_2 P = x_1, y_1$ and $r = x_1 \ mod \ n$, if $r = 0$ then return to step (a).
   (c) Compute $k_2^{-1} \ mod \ n$;
   (d) Compute $s = k_2^{-1}(SHA(m) + k_1 r) \ mod \ n$, if $s = 0$ then return to step (a);
   (e) The signature for the message m is (r, s).
(3) **Signature Verification**
   (a) The second user, after receiving the signature (r, s) computes $c = s^{-1} \ mod \ n$ and $SHA(m)$;
   (b) Then computes $u_1 = H(m)c \ mod \ n$ and $u_2 = rc \ mod \ n$;
   (c) Then computes $u_1 P + u_2 Q = (x_0, y_0)$ and $v = x_0 \ mod \ n$;
   (d) If $v = r$ then the signature is valid.

The ECDSA advantages and disadvantages can be studied in [9]. After the verification, the user and the server have to generate a secret key for the encryption. For the protocol presented in [1], generating third key is made through a scalar addition, while in others is used a public-key algorithm. Studying these two protocols we chose a zero knowledge authentication for our protocol. We will motivate this choice in 3. Aydos et al. proposed an elliptic curve authentication protocol in [1]. The protocol uses ECDSA for the authentication and the Diffie-Hellman key exchange scheme to establish session key.

ECDH. Let $P(x, y)$ be a point of order n on the elliptic curve E defined over F_p. The ECDH has the following steps:

(1) A generates a random number $d_A \in [2, n-2]$, then computes $D_A = d_A P$ and sends it to B;
(2) B generates a random number $d_B \in [2, n-2]$, then computes $D_B = d_B P$ and sends D_B to A;
(3) A computes the key $sk_A = d_A D_B$ and B computes the key $sk_B = d_B D_A$;
(4) The two results represent the common key because $sk_A = sk_B$.

The entire protocol can be studied in [1]. The protocol has significant superiorities in terms of speed, storage requirement and bandwidth requirement. In spite of all these advantages, the protocol has be proven to be vulnerable to man-in-the-middle attack [24]. A protocol based on Aydos' protocol is the Mangipudi et al.'s protocol. In this protocol the user needs to authenticate himself to the server. The protocol, like the Aydos' protocol, has two phases: the initialization phase and the user authentication phase. Unlike the Aydos' protocol where in the first phase the CA (Certificate Authority) sends its own public key, in the Mangipudi's protocol the CA sends servers public key and expiration time. The Mangipudi et al.'s protocol is robust to man-in-the-middle attack, but it's vulnerable to the forging certificate attack launched by attacker after he has forged the certificate. An example of such an attack was demonstrated in [24].

3. Our Method

In the section above we have presented several types of authentication protocols based on elliptic curve cryptography. Some of them are suitable for special situations, while others have been proven to be vulnerable to various attacks. We propose an authentication protocol suitable for group communication. Suppose that maximum 30 persons want to communicate simultaneously. They have to validate a very important document. Being an important decision it can be taken only if all the communicating parties agree. If one person disagrees, the document will not be validated, even if all the others want to validate it. In such cases it is very important to communicate in real time. If one of the parties is not on-line, the server will not allow the others to take any decision. To avoid an impersonation attack, the server will stop transmitting any kind of messages (even if they are not about the decision that the group has to take) if one of the parties is not on-line. The server will allow the users to stay on-line (but not to communicate) as long as they want. Along with every attempt to communicate the users will receive a denial message from the server. This message will, also, contain the nickname of the missing person (or persons). This will allow the others to find a solution faster. When all the users are logged the server will send a message to every user to announce that they can start the communication. If the communication is started and during it, one of the users signs off, the process will be stopped immediately. This kind of protocol is very useful for big companies where shareholders are very busy and cannot attend all the meetings. Some of these meetings can be delayed depending on the shareholders' schedules. But, the meetings' whose delaying may cause great loss of money, must be avoided. An efficient solution to this problem is using communicating technology in real time. This technology must have a high level of security because the transmitted information must be kept secret. Using an authentication protocol is the easier and the cheaper way to solve the problem.

3.1. Preliminaries. Most of the authentication protocols based on elliptic curve cryptography use the ECDSA. This is because it provides a high level of security. For our protocol, we chose not to use the Elliptic Curve Digital Signature Algorithm because the communication starts after all the parties are authenticated, and authenticating 30 persons can take several minutes. We chose to use zero knowledge to authenticate the users. The purpose of zero knowledge protocols is to prove the knowledge of a secret without revealing it. A zero knowledge protocol must satisfy the followings:

(1) **Completeness** if the statement is true, the verifier will be convinced of this fact by an honest prover.
(2) **Soundness** if the statement is false, no cheating prover can convince the verifier that it is true.
(3) **Zero-knowledge** if the statement is true, no cheating verifier learns anything other than this fact.

Every user from the group has a secret information. Each one has to prove that he knows the information (without revealing it) to the server. So, the prover is the user and the verifier is the server. Of course, the secret information of each user is different. The server will identify each user through a demonstration of his knowledge. Our protocol does not reveal any detail about the secret information. So, neither the server nor an eavesdropper will not obtain the user's secret, avoiding an impersonation of the user. The main idea of the zero knowledge authentication is that the verifier

asks the prover a question (or questions) related to the secret information, but in a manner that the answer does not reveal the secret or even a part of it. So, the verifier and the prover exchange messages which will convince the verifier if the prover is or is not who he pretends to be.

One of the most popular zero knowledge protocols is Schnorr's protocol. The protocol proves the knowledge of a discrete logarithm and it is defined over a cyclic group G_q of order q with generator g. To prove that $x = \log_g y$ the steps below will be followed:

(1) The prover choses a random number r and computes $t = g^r$ and sends it to the verifier;
(2) The verifier choses a random number c and sends it to the prover;
(3) The prover sends $s = r + cx$;
(4) The verifier accepts the prover if $g^s = ty^c$.

Schnorr's protocol can be studied in [21]. Our protocol uses elliptic curve version of Schnorr's zero knowledge protocol [22]. Schnorr's protocol with elliptic curve is described below.

Before the protocol is started we set the parameters: (q, a, b, P, n, h) where q specifies the finite field F_q, a and b define the elliptic curve, P is a point on the curve of order n, and h is the cofactor. We note the prover's secret information with α and the user makes public $Z = \alpha P$.

(1) The prover choses a random number r, computes $X = rP$ and sends X to the verifier;
(2) The verifier choses a random number e and sends it to the prover;
(3) The prover computes $y = (\alpha e + r)\ mod\ n$ and sends y to the verifier;
(4) The verifier accepts the prover if $yP + eZ = X$.

3.2. Protocol. The proposed one is based on ECDH protocol principles and use ECDSA in the signing phases. To implement this protocol we have to make some assumptions first:

- every user is connected to a server;
- the public keys are saved in a public file;
- the server has a pair of keys, too;
- the server key is the only one that does not need to be verified.

Authentication.

(1) $user \longrightarrow server : sends_{serv}(X, N_1)$
(2) $server \longrightarrow user : sends_{user}(e, N_1)$
(3) $user \longrightarrow server : sends_{serv}(y)$
(4) $server \longrightarrow accept/reject$

First the user sends $X = rP$ and a random message N_1, both encrypted with the server's key. The server decrypts the stream received and finds N_1. Then, the server sends the random number e and N_1 encrypted with the user key. When the user receives the stream, he will know that it is send from the server because only the server knew the random message N_1. He, then, computes $y = (\alpha e + r)\ mod\ n$, encrypts it with the server's key and sends the result to the server. If $yP + eZ = X$ the server accepts the user, else a rejection is made. These steps are repeated for every user.

Communication Process. When all the users are authenticated the communication can start. All the users have the same key pair (Pub, Sec) where Pub is the public key and Sec is the secret one. These keys along with the server's ones are generated every time a communication session is started. All the users use the same keys because

they all have to know the messages sent from any user of the group. This will avoid conflicts and plotting. Two users cannot communicate through this protocol without the knowledge of the others. This is very important because, like we said earlier, they all have to decide on the document. So, if one user receives a message it can be read by all the others, because they all can decrypt it. On the other hand, if one user's key is found by an intruder all the users are affected. The keys can be established using an elliptic curve protocol which keeps the point from the elliptic curve private, for example the one proposed in [7], based on the elliptic curves computations described in [6]. Such a protocol is described in [11]. By keeping the point $P \in E$ private, the security level is increased. The only public parameters are the prime number n, and a, b defining the curve $E_n(a, b) : y^2 = x^3 + ax + b$ where $gcd(4a^3 + 27b^2, n) = 1$. The key pair (e, d) is established by following the RSA algorithm [18]. Suppose the two communicating parties are Alice and Bob. Alice choses X_A a random number in F_n, R_A another random number in F_n and P_A a point on the curve. All these three values are secret. Similarly, Bob choses X_B, R_B and P_B. The steps of the algorithm are:

(1) Alice computes $G_A = X_A P_A$ and send it to Bob;
(2) Bob computes $G_B = X_B P_B$ and send it to Alice;
(3) Alice computes $S_A = R_A G_B$ and send it to Bob;
(4) Bob computes $S_B = R_B G_A$ and send it to Alice;
(5) Alice computes the session key $Pub = e(S_A + S_B)$;
(6) Bob computes the session key $Pub = e(S_A + S_B)$.

Multiplying by e gives the protocol public key characteristics. So, the private key will be $Sec = d(S_A + S_B)$. In the same time, the multiplication provides an increased security, so that the protocol will not suffer from the man-in-the-middle attack. As we can see, the method has only two pass key agreements, which mean no communication overhead will be added.

3.3. Comparison. You have probably already noticed that the authentication phase is the most important one. This is because the messages have to be known by all the group's members and so if an intruder succeeds to sign in all the information is compromised. We did not use a digital signature algorithm because there were much more operations. The secret knowledge provides an authentication with three times fewer steps and at least the same level of security. We applied an encryption to the result obtained through Schnorr's protocol to increase the security. To compute the scalar multiplication we recommend the Montgomery's Point Multiplication Method where is used a combination of point-additions and point-doublings. This protocol can be implemented without using elliptic curve cryptography. In this case we will use the classic Schnorr's protocol for authentication and for encryption the most efficient method is RSA. But, the keys used for encrypting and decrypting the messages must be significantly larger to reach the same level of security. To avoid a high complexity the proposed protocol uses elliptic curve cryptography. Also, the elliptic curve cryptography provides the security needed in such a sensitive situation. We call it a sensitive situation because all the users have the same keys for one session. If one key is compromised all the other users are compromised, too. We can avoid this situation by using different keys for each user. We did not choose to do that because it is more important that all messages can be read by any member of the group.

4. Conclusions

We presented a relatively simple protocol for a group communication based on elliptic curve cryptography. The protocol has a low complexity mainly because the group's members have the same key pairs, but also because the authentication is made through zero knowledge. Using elliptic curve cryptography provides a methodology for obtaining high-speed implementations of authentication protocols and encrypted message techniques while using fewer bits for the keys. For establishing the encryption/decryption keys we chose a method were no point on the elliptic curve is made public. Keeping the elliptic curve point private increases the security of the algorithm. This method is also easy to implement, being based on the characteristics elliptic curves and RSA encryption systems.

References

[1] M. Aydos, B. Sunar and C. K. Koc, An Elliptic Curve Cryptography based Authentication and Key Agreement Protocol for Wireless Communication, *Proceedings of the 2nd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, Dallas (1998), 1–12.

[2] C. Boyd and A. Mathuria, *Protocols for authentication and Key Establishment*, Springer-Verlag, 2003.

[3] Certicom Research, SEC 2: Recommended Elliptic Curve Domain Parameters, Standards for efficient Cryptography, Version 1.0, Sept. (2000).

[4] K. Chalkias, G. Filiadis and G. Stephanides, Implementing Authentication Protocol for Exchanging Encrypted Messages via an Authentication Server based on Elliptic Curve Cryptography with the El Gamals Algorithm, *IEC* (Prague), (2005), 137–142.

[5] N. Constantinescu, Authentication ranks with identities based on elliptic curves, *Annals of the University of Craiova, Mathematics and Computer Sciences Series* **34** (2007), 94–99.

[6] N. Constantinescu, Elliptic curve cryptosystems and scalar multiplication, *Annals of the University of Craiova, Mathematics and Computer Sciences Series* **37** (2010), no. 1, 27–34.

[7] N. Constantinescu, The GN-authenticated key agreement, *Journal of Applied Mathematics and Computation, Elsevier* **170** (2005), no. 1, 531–544.

[8] U.S. Dept of Commerce/NIST, Digital Signature Standard (DSS), FIPS PUB 186-2, Jan. (2000).

[9] IEEE P1363. Standard specifications for public-key cryptography. Draft version 7, Sept. (1998).

[10] D. Hankerson, A. Menezes and S. Vastone, *Guide to Elliptic curve cryptography*, Springer-Verlag, 2003.

[11] K. Kaabneh and H. Al-Bdour, Key Exchange Protocol in Elliptic Curve Cryptography with No Public Point, *American Journal of Applied Sciences* **2** (2005), no. 8, 1232–1235.

[12] N. Koblitz, A. Menezes and S. Vanstone, The state of elliptic curve cryptography, *Designs, Codes and Cryptography* **19** (2000), no. 2-3, 173–193.

[13] N.Koblitz, Elliptic curve cryptosystems, *Mathematics of Computation* **48** (1987), no. 1, 203–209.

[14] N. Koblitz, CM Curves with Good Cryptographic Properties, *Proc. Crypto'91*, Springer-Verlag (1992), 279–287.

[15] V. Muller, Fast Multiplication on Elliptic Curve over Small Field of Characteristic Two, *Journal of Cryptology* **11** (1998), no. 2, 219-234.

[16] NIST, Special Publication 800-57: Recommendation for Key Management. Part 1: General Guideline, Draft Jan. (2003).

[17] M. Rosing, *Implementing Elliptic Curve Cryptography*, Manning, 1999.

[18] W. Sallings, *Cryptography and Network Security*, Prentice Hall, Third Ed, 2003.

[19] A. Saxena and B. Soh, An Authentication Protocol For Mobile Agents Using Bilinear Pairings, Dept. of Computer Science and Computer Engineering La Trobe University, Bundoora, VIC, Australia 3086, 1 Sept. (2005).

[20] N. Smart, How secure are elliptic curves over composite extension fields?, *EUROCRYPT 2001*, *LNCS 2045* Springer-Verlag (2001), 30–39.

[21] C. P. Schnorr, Efficient identification and signatures for smart cards, *G Brassard, ed. Advances in Cryptology - Crypto '89*, Springer-Verlag (1990), 239–252.
[22] C. P. Schnorr, Efficient signature generation by smart cards, *Journal of Cryptology* **4** (1991), no. 3, 161–174.
[23] L. C. Washington, *Elliptic curves Number Theory and Cryptography*, Chapman&Hall /CRC, 2003.
[24] L. Yongliang, W. Gao, H. Yao, and X. Yu, Elliptic Curve Cryptography Based Wireless Authentication Protocol, *International Journal of Network Security* **5** (2007), no. 3, 327-337.

(Nicolae Constantinescu) Faculty of Mathematics and Computer Science, University of Craiova, Al.I. Cuza Street, No. 13, Craiova RO-200585, Romania, Tel. & Fax: 40-251412673
E-mail address: nikyc@central.ucv.ro