

# ECC Based Convertible Authenticated Encryption Scheme Using Self-Certified Public Key Systems

Tzong-Sun Wu<sup>a</sup> and Han-Yu Lin<sup>b</sup>

<sup>a</sup> Department of Computer Science and Engineering  
National Taiwan Ocean University  
Keelung, 202, Taiwan

<sup>b</sup> Department of Computer Science  
National Chiao Tung University, Hsinchu, 300, Taiwan  
hanyu.cs94g@nctu.edu.tw

## Abstract

This paper presents an ECC (Elliptic Curve Cryptography) based convertible authenticated encryption (CAE) scheme using self-certified public key systems. Combining the merits of self-certified public key cryptosystems, the proposed scheme has the property that authenticating the public key and verifying the signature can be simultaneously carried out within one step, which helps reducing computation efforts. Based on the ECC, our scheme can provide crucial benefits to those applications of mobile devices with the limited computing power and insufficient storage space for its shorter key length. The proposed scheme also satisfies the security requirement of computational secrecy which ensures that the generated authenticated ciphertext is computationally indistinguishable with respect to two candidate messages. Moreover, the designated recipient has the ability to prove himself, if needed, to anyone that he is the actual recipient.

**Keywords:** elliptic curve cryptography, convertible authenticated encryption, self-certified public key

## 1. Introduction

In 1976, Diffie and Hellman [2] introduced the first public key system based on the discrete logarithm problem (DLP). Each user in the system owns a private

key and its corresponding public key which is accessible to anyone. With the two keys, one can perform the public key encryption or digital signature schemes [3, 10]. However, one should always take care that the authenticity of obtained public keys since a malicious adversary might plot an active attack by substituting a fake public key for the genuine one. In 1991, Girault [4] proposed a self-certified public key system in which the public key validation can be combined with other subsequent cryptographic mechanisms like the signature verification. That is, the tasks of authenticating the public key and verifying the signature can be simultaneously achieved in one step.

The Elliptic Curve Cryptography (ECC) first introduced by Koblitz [6] and Miller [8] is especially applicable to the applications of mobile devices with the limited computing power and insufficient storage space. A significant characteristic of the ECC is that the key length is shorter than that of the conventional cryptography under the same level of security, which helps faster execution and more bandwidth savings.

To meet requirements of some E-Commerce applications like credit card transactions that digital signatures must simultaneously satisfy the need of confidentiality, an authenticated encryption scheme was proposed by Horster *et al.* [5] in 1994. Such scheme allows a signer to generate an authenticated ciphertext but only the designated recipient can recover the message and verify its corresponding signature for the purpose of confidentiality. Yet, a later dispute that the signer disclaims having generated a signature might occur. To convince anyone of the signer's dishonesty, the designated recipient must have the ability to convert the signature into an ordinary one for protecting his rights or benefits. In 2002, Wu and Hsu [11] proposed a convertible authenticated encryption (CAE) scheme in which the signature conversion process is rather simple and can be solely done by the recipient without any computation efforts. However, the Wu-Hsu scheme cannot provide the computational secrecy, i.e., the ciphertext is computationally distinguishable with respect to two candidate messages. To eliminate such security weakness, in this paper, we will propose an ECC based CAE scheme with computational secrecy combining the merits of self-certified public key cryptosystems. In addition, the designated recipient also has the ability to prove himself, if needed, to anyone that he is the actual recipient.

## 2. Self-Certified CAE Scheme Based on the ECC

In this section, we present a self-certified CAE scheme based on the ECC. The proposed scheme can be divided into four stages: the user registration, the signature generation and verification, the signature conversion, and the recipient proof stages. Initially, the system authority (SA) chooses the following necessary parameters:

$p$ : a large prime;

- $a, b$ : two parameters in  $Z_p$  satisfying that  $4a^3 + 27b^2 \pmod p \neq 0$ ;  
 $E_p(a, b)$ : an elliptic curve over  $\text{GF}(p)$  containing a set of points  $(x, y)$  satisfying that  $y^2 = x^3 + ax + b \pmod p$ ;  
 $O$ : a point at infinity over  $E_p(a, b)$ ;  
 $G$ : the base point of order  $q$  over  $E_p(a, b)$ , where  $q$  is a large prime;  
 $h(\cdot)$ : a secure one-way hash function which accepts input of various length and generates output of a fixed length; note that the input of a point over  $E_p(a, b)$  represents the input of the concatenation of the  $x$ - and  $y$ - coordinates of that point;  
 $\gamma$ : the SA's private key for  $\gamma \in Z_q^*$ ;  
 $B$ : the SA's public key computed as
- $$B = \gamma G \quad \text{over } E_p(a, b). \quad (1)$$

All of the above parameters are made public except for the SA's private key  $\gamma$ . In the following, all elliptic curve point operations are manipulated over  $E_p(a, b)$ . Details of each stage are shown as follows:

**The user registration stage:** To join the system, each user  $U_i$  associated with the identifier  $ID_i$  performs the interactive registration process with the SA.

**Step 1**  $U_i$  first chooses an integer  $t_i \in Z_q^*$  to compute

$$V_i = h(t_i, ID_i)G, \quad (2)$$

and then delivers  $(V_i, ID_i)$  to the SA.

**Step 2** Upon receiving  $(V_i, ID_i)$ , the SA chooses an integer  $z_i \in Z_q^*$  to compute

$$Y_i = (h(ID_i)^{-1} \pmod q)(V_i + z_i G), \quad (3)$$

$$w_i = z_i + h(Y_i, ID_i)\gamma \pmod q, \quad (4)$$

and returns  $(Y_i, w_i)$  to  $U_i$ .

**Step 3**  $U_i$  computes  $x_i$  as

$$x_i = w_i + h(t_i, ID_i) \pmod q, \quad (5)$$

and checks its validity with Eq. (6).

$$h(Y_i, ID_i)B + h(ID_i)Y_i \stackrel{?}{=} x_i G. \quad (6)$$

If the above equation holds,  $U_i$  accepts  $(x_i, Y_i)$  as his private and public keys. Theorem 1 proves the correctness of Eq. (6).

**Theorem 1.**  $U_i$  can perform Eq. (6) to authenticate the public key  $Y_i$  with respect to his private key  $x_i$ .

**Proof:** From the left-hand side of Eq. (6), we have

$$\begin{aligned}
h(Y_i, ID_i)B + h(ID_i)Y_i &= h(Y_i, ID_i)B + (V_i + z_iG) && \text{(by Eq. (3))} \\
&= (h(Y_i, ID_i)\gamma + z_i)G + V_i && \text{(by Eq. (1))} \\
&= w_iG + V_i && \text{(by Eq. (4))} \\
&= (w_i + h(t_i, ID_i))G && \text{(by Eq. (2))} \\
&= x_iG && \text{(by Eq. (5))}
\end{aligned}$$

which equals to the right-hand side of Eq. (6).

Q.E.D.

**The signature generation and verification stage:** When  $U_a$  wants to send  $U_b$  an authenticated ciphertext for the message  $m$  with embedded redundancy.  $U_a$  first chooses an integer  $k \in Z_q^*$  and computes the following parameters:

$$C = k(h(Y_b, ID_b)B + h(ID_b)Y_b) \quad (7)$$

$$r_1 = mh(C)^{-1} \bmod p \quad (8)$$

$$r_2 = h(m, h(kG), C) \bmod q \quad (9)$$

$$s = k(1 + x_a r_2)^{-1} \bmod q \quad (10)$$

Here,  $(r_1, r_2, s)$  is the signature for  $m$ , and is then sent to the designated recipient  $U_b$ . After receiving the signature,  $U_b$  first computes  $K$  and  $C'$  from Eqs. (11) and (12), respectively.

$$K = s(r_2(h(Y_a, ID_a)B + h(ID_a)Y_a) + G) \quad (11)$$

$$C' = x_b K \quad (12)$$

The message  $m$  with the embedded redundancy can be recovered from Eq. (13).

$$m = h(C')r_1 \bmod p \quad (13)$$

Afterward  $U_b$  can verify the signature  $(r_1, r_2, s)$  by testing Eq. (14).

$$r_2 = h(m, h(K), C') \bmod q \quad (14)$$

If it holds, the signature is valid; meanwhile, the signer's public key  $Y_a$  is simultaneously authenticated. We demonstrate that Eqs. (13) and (14) work correctly as the proofs of Theorems 2 and 3, respectively.

**Theorem 2.** The recipient  $U_b$  can recover the message  $m$  with the embedded redundancy with Eq. (13).

**Proof:** From the right-hand side of Eq. (13), we have

$$\begin{aligned}
&h(C')r_1 \\
&= h(x_b K)r_1 && \text{(by Eq. (12))} \\
&= h(sx_b(r_2(h(Y_a, ID_a)B + h(ID_a)Y_a) + G))r_1 && \text{(by Eq. (11))} \\
&= h((sx_b)(r_2x_aG + G))r_1 && \text{(by Eq. (6))}
\end{aligned}$$

$$\begin{aligned}
 &= h(sx_a r_2 x_b G + sx_b G) r_1 \\
 &= h((sx_a r_2 + s)(x_b G)) r_1 \\
 &= h((sx_a r_2 + s)(h(Y_b, ID_b)B + h(ID_b)Y_b)) r_1 && \text{(by Eq. (6))} \\
 &= h(k(h(Y_b, ID_b)B + h(ID_b)Y_b)) r_1 && \text{(by Eq. (10))} \\
 &= h(C) r_1 && \text{(by Eq. (7))} \\
 &= m \pmod{p} && \text{(by Eq. (8))}
 \end{aligned}$$

which equals to the left-hand side of Eq. (13).

Q.E.D.

**Theorem 3.** A valid signature  $(r_1, r_2, s)$  should satisfy Eq. (14) which also authenticates the public key  $Y_a$ .

**Proof:** From the right-hand side of Eq. (14), we have

$$\begin{aligned}
 &h(m, h(K), C') \\
 &= h(m, h(K), x_b K) && \text{(by Eq. (12))} \\
 &= h(m, h(sG + sr_2(h(Y_a, ID_a)B + h(ID_a)Y_a)), \\
 &\quad x_b(sG + sr_2(h(Y_a, ID_a)B + h(ID_a)Y_a))) && \text{(by Eq. (11))} \\
 &= h(m, h((sr_2)x_a G + sG), x_b((sr_2)x_a G + sG)) && \text{(by Eq. (6))} \\
 &= h(m, h((sx_a r_2 + s)G), x_b(sx_a r_2 + s)G) \\
 &= h(m, h(kG), x_b kG) && \text{(by Eq. (10))} \\
 &= h(m, h(kG), k(h(Y_b, ID_b)B + h(ID_b)Y_b)) && \text{(by Eq. (6))} \\
 &= h(m, h(kG), C) && \text{(by Eq. (7))} \\
 &= r_2 \pmod{q} && \text{(by Eq. (9))}
 \end{aligned}$$

which equals to the left-hand side of Eq. (14).

Q.E.D.

**The signature conversion stage:** To deal with the case of a later dispute, the recipient  $U_b$  can just release the recovered message  $m$  and the converted signature  $(r_2, s, C')$ . Anyone can first compute  $K$  from Eq. (11.) and then validate the signature with Eq. (14). If the checking of Eq. (14) holds, he assures that the signature is generated by  $U_a$ .

**The recipient proof stage:** For convincing someone, say,  $U_c$ , that he is the real recipient, the recipient  $U_b$  can perform the following interactive steps with  $U_c$ :

**Step 1**  $U_b$  sends the converted signature  $(r_2, s, C')$  to  $U_c$ .

**Step 2**  $U_c$  first computes  $K$  by Eq. (11) and then checks the signature's validity with Eq. (14). If it holds,  $U_c$  proceeds to the next step; otherwise, the protocol is terminated.

- Step 3**  $U_c$  randomly chooses an integer  $e$  to compute  $E = eK$  and then transmits  $E$  to  $U_b$ .
- Step 4** Upon receiving  $E$ ,  $U_b$  computes  $W = x_bE$  and returns it to  $U_c$ .
- Step 5**  $U_c$  computes  $W' = eC'$  and checks whether  $W = W'$ . If it holds,  $U_c$  is convinced that  $U_b$  is the designated recipient.

### 3. Security Considerations and Performance Evaluation

In this section, we will discuss some security considerations of the proposed scheme followed by the performance evaluation.

#### 3.1 Security Considerations

The subsection demonstrates that the proposed scheme is secure against known active attacks. The mathematical assumptions of our proposed scheme are the elliptic curve discrete logarithm problem (ECDLP) [6] and the one-way hash function (OHF) [2, 6, 8]. We analyze the security considerations from three perspectives: confidentiality, unforgeability and non-repudiation.

##### *Confidentiality*

- (i) ***The confidentiality of the SA's private key:*** To directly derive the SA's private key  $\gamma$  from Eq. (1), the attacker will face the intractability of the ECDLP which is computationally infeasible to invert. Further, if he attempts to derive the SA's private key  $\gamma$  from Eq. (4) by sending a registration request, he has to know the secret integer  $z_i$  which is protected by the ECDLP as well.
- (ii) ***The confidentiality of the user  $U_i$ 's private key:*** An attacker cannot successfully retrieve the user  $U_i$ 's private key  $x_i$  from Eq. (5) unless he has the ability to invert the ECDLP for obtaining the secret value  $h(t_i, ID_i)$  from Eq. (2).
- (iii) ***The confidentiality of the common key between  $U_a$  and  $U_b$ :*** Upon intercepting a valid signature  $(r_1, r_2, s)$ , an attacker may try to derive  $U_a$  and  $U_b$ 's common key  $Y_{ab}(= x_bY_a = x_aY_b)$  from Eqs. (11) and (12). However, he will face the intractability of the ECDLP.
- (iv) ***The confidentiality of the signing message:*** To recover the message  $m$ , the attacker has to retrieve the common key  $Y_{ab}$  between  $U_a$  and  $U_b$  first. According to the above analysis of the confidentiality of the common key  $Y_{ab}$ , however, he cannot successfully plot the attack under the protection of the ECDLP.

- (v) **The semantic security considerations:** Consider the adversary attempting to guess the correct message from two candidate ones with respect to the ciphertext  $\sigma'$ . He may proceed to perform the validation equality, Eq. (14), and guess the one satisfying the equality. To complete the task, however, he needs to compute an additional parameter  $C'$  which is protected by the recipient's private key  $x_b$ . Thus, he has the advantage of winning the game with no more than  $1/2$ , i.e., the produced authenticated encryption message is computationally indistinguishable.

### **Unforgeability**

- (i) **The unforgeability of the public key:** It is computationally infeasible for an attacker to forge a valid public key satisfying Eq. (6) under the security assumptions of the ECDLP and the OHF.
- (ii) **The unforgeability of the resulted signature:** Forging a valid signature  $(r''_1, r''_2, s'')$  on an arbitrarily chosen message  $m''$ , an attacker may first randomly choose  $(r''_2, s'')$  to compute  $K''$  fulfilling Eq. (11) and then arbitrarily choose  $C''$  to derive  $r''_1$  satisfying Eq. (13). However, the randomly chosen  $(r''_2, s'')$  cannot pass the test of Eq. (14). Moreover, based on the intractability of the ECDLP, he cannot derive the signer's private key to forge a valid signature either.
- (iii) **The unforgeability of the converted signature:** To forge a valid converted signature  $(r''_2, s'', C'')$ , an attacker may first choose a random  $C''$  and then computes  $(r''_2, s'')$  satisfying Eq. (14). Unfortunately, he cannot make it under the protection of the ECDLP. Similarly, choosing  $(r''_2, s'')$  first to compute  $C''$  for satisfying Eq. (14) is not workable.

### **Non-repudiation**

The resulted signature  $(r_1, r_2, s)$  generated by the signer  $U_a$  can only be verified by the designated recipient  $U_b$ . In case of a later dispute, the designated recipient  $U_b$  can announce the converted signature  $(r_2, s, C')$  to convince anyone that the signature is indeed generated by the signer  $U_a$ . According to the analyses of the confidentiality of the user  $U_i$ 's private key and the unforgeability of the resulted signature, any attacker cannot forge a valid signature without knowing the user  $U_i$ 's private key  $x_i$ . Therefore, the signer  $U_a$  cannot deny his signatures.

From the above discussions, it can be seen that our proposed scheme is secure against known active attacks even under the semantic security based on the hardness of ECDLP and the OHF assumptions.

## **3.2 Performance Evaluation**

To facilitate the reading, we first define the following notations which will be used in the article later.

- $T_h$ : the time for performing a one-way hash function  $h$   
 $T_m$ : the time for performing a modular multiplication computation  
 $T_i$ : the time for performing a modular inverse computation  
 $T_{EC-a}$ : the time for performing a modular addition computation over an elliptic curve  
 $T_{EC-m}$ : the time for performing a modular multiplication computation over an elliptic curve

We ignore the time for performing the modular addition because it is negligible as compared to computing time of performing other operations. Mitchell *et al.* [9] also stated that a hash function would not take longer time than that of a modular multiplication computation. Consequently, we can use one  $T_m$  to approximate one  $T_h$  without affecting the correctness. The detailed performance evaluation of the proposed scheme is shown as Table 1.

Table 1. Performance analyses of the proposed scheme

| Stages                                | Time complexity |   | Rough Estimation <sup>‡</sup> |
|---------------------------------------|-----------------|---|-------------------------------|
| User registration                     | $U_i$           | $3T_h + T_{EC-a} + 4T_{EC-m}$               | $119.12T_m$                   |
|                                       | SA              | $2T_h + T_i + T_m + T_{EC-a} + 2T_{EC-m}$   | $64.12T_m$                    |
| Signature generation and verification | $U_a$           | $5T_h + 2T_i + 3T_m + T_{EC-a} + 4T_{EC-m}$ | $130.12T_m$                   |
|                                       | $U_b$           | $5T_h + T_m + 2T_{EC-a} + 5T_{EC-m}$        | $151.24T_m$                   |
| Signature conversion                  | $U_b$           | 0   |                               |
| Recipient proof                       | $U_b$           | $T_{EC-m}$                                  | $29T_m$                       |
|                                       | $U_c$           | $4T_h + 2T_{EC-a} + 4T_{EC-m}$              | $120.24T_m$                   |

Remark: <sup>‡</sup>  $T_h \approx T_m$ ,  $T_i \approx 3T_m$ ,  $T_{EC-a} \approx 0.12T_m$  and  $T_{EC-m} \approx 29T_m$  [7].

#### 4. Conclusions

In this paper, we have proposed an ECC based self-certified CAE scheme with computational secrecy. Combining with the merits of self-certified public key cryptosystems, the proposed scheme can greatly reduce computation efforts for that the tasks of verifying the signature and authenticating the public key can be simultaneously carried out in one step. The computational secrecy also ensures that the generated ciphertext is indistinguishable with respect to two candidate messages. In case of a later dispute, the recipient has the ability to convert the

signature into an ordinary one without any extra overheads. Moreover, the significant characteristic of the shorter key length of the ECC makes our scheme suitable for those applications of mobile devices with the limited computing power and insufficient storage space like cell phones, etc.

## References

- [1] S. Araki, S. Uehara and K. Imamura, The limited verifier signature and its application, *IEICE Transactions on Fundamentals* **E82-A** (1) (1999) 63-68.
- [2] W. Diffie and M. Hellman, New directions in cryptography, *IEEE Transactions on Information Theory* **IT-22** (6) (1976) 644-654.
- [3] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, *IEEE Transactions on Information Theory* **IT-31** (4) (1985) 469-472.
- [4] M. Girault, Self-certified public keys, *Advances in Cryptology – EUROCRYPT’91*, Springer-Verlag, 1991, pp. 491-497.
- [5] P. Horster, M. Michel, H. Peterson, Authenticated encryption schemes with low communication costs, *Electronics letters* **30** (15) (1994) 1212-1213.
- [6] N. Koblitz, Elliptic curve cryptosystems, *Mathematics of Computation* **48** (177) (1987) 203-209.
- [7] N. Koblitz, A. Menezes, S. Vanstone, The state of elliptic curve cryptography, *Designs, Codes and Cryptography* **19** (2-3) (2000) 173-193.
- [8] V. Miller, Use of elliptic curves in cryptography, *Advances in Cryptology – CRYPTO’85*, Springer-Verlag, 1985, pp. 417-426.
- [9] C.J. Mitchell, F. Piper and P. Wild, Digital signature, In: Simmons, G.J. (Ed.), *Contemporary Cryptology: The Science of Information Integrity*, IEEE Press, 1992, pp. 325-378.
- [10] R. Rivest, A. Shamir and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, *Communications of the ACM* **21** (2) (1978) 120-126.
- [11] T.S. Wu and C.L. Hsu, Convertible authenticated encryption scheme, *The Journal of Systems and Software* **62** (3) (2002) 205-209.

**Received: August 4, 2007**