Journal of Computers Vol. 27, No. 3, 2016, pp. 111-122
doi:10.3966/199115592016102703011

Novel Electronic Check Mechanism Using Elliptic Curve Cryptosystem

Chin-Chen Chang[1,2,3,*] Shih-Chang Chang[1] Yu-Ching Wu[1]

[1] Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan, 621, R.O.C.
alan3c@gmail.com

[2] Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan, 40724, R.O.C.

[3] Department of Computer Science and Information Engineering, Asia University, Taichung 41354, Taiwan

Received 16 September 2011; Revised 19 February 2012; Accepted 20 February 2012

Abstract. The electronic check (e-check) application is a significant issue in electronic commerce. Chaum first proposed the electronic check mechanism in 1988, after which methods based on RSA or ElGamal improved the process. However, these e-check methods use exponential operations to meet security requirements, so they have a heavy computational cost. In order to reduce the computational cost, we propose a novel electronic check mechanism using an elliptic curve cryptosystem. Our proposed mechanism meets the security requirements of electronic checks and has low computational costs compared to some other methods.

Keywords: electronic check, elliptic curve cryptosystem, mutual authentication, user privacy, security

1 Introduction

Many e-commerce applications have been implemented in the last two decades, one of the best known of which is the electronic payment mechanism. Electronic payment mechanisms can be classified into three types: on-line credit cards, electronic cash, and electronic checks [1-6]. When the payer uses an electronic credit card as the payment tool, the whole payment process must be done online to ensure security. When the payer uses electronic cash (e-cash) to pay bills on the Internet, the electronic cash works and transacts just like cash. When the payer uses an electronic check (e-check) as the payment tool, the check works like a paper check. However, all three of these mechanisms have limitations in the amount that can be transacted. This limitation is a particular problem with electronic credit cards, where large transactions are frequent. While e-cash is suitable for micro-payments, it has a weakness in that anyone can obtain e-cash and use it to make a transaction. Therefore, the electronic check has become a major player and an important part of the electronic payment mechanism.

The first proposed e-check mechanism was introduced by Chaum in 1988 [7], but many new e-check mechanisms have since been developed, most of them using Public-key Cryptography, like the Rivest, Shamir and Adleman [8] or ElGamal [9]. In 1985, Koblitz [10] and Miller [11] proposed a new type of public-key cryptography [12] based, for the first time, on the elliptic curve. The security of the elliptic curve is based on the difficulty of computing the Elliptic Curve Discrete Logarithm Problem (ECDLP). Another advantage of the elliptic curve is the size of the key; under the same security environment assumption, the key size of 160 bits in the elliptic curve can be equal to the key size of 1024 bits in Rivest, Shamir and Adleman. Hence, the elliptic curve is more suitable for cryptosystems and lowers the computation cost.

* Corresponding Author

In this paper, we propose a novel electronic check mechanism using an elliptic curve cryptosystem. Our mechanism not only meets the requirements of electronic checks but also has low computational cost and can resist attacks on electronic payment mechanisms.

Any new e-check mechanism must ensure user privacy, mutual authentication, uniqueness, robustness and non-repudiation [13].

1. **User Privacy:** To ensure privacy, the customer can use a virtual identity to make a transaction with the merchant, and the merchant does not have to know who used the e-check.
2. **Mutual Authentication:** Both the customer and the merchant can authenticate each other, which provides additional security in the payment mechanism.
3. **Uniqueness:** The bank uses the customer's identity to produce the e-checkbook; since the identity of the customer is unique, the same e-checkbook cannot be generated for other customers. The bank can also use the relationship to find corresponding data to verify the e-check.
4. **Robustness:** The e-checkbook can be generated only by the valid customer and the trusted bank. Since the transaction is often complicated, this requirement helps prevent a malicious attacker from forging the e-check.
5. **Non-Repudiation:** When a customer uses an e-check for a transaction, the customer cannot claim that he has not done so.

The rest of this article is organized as follows. In Section 2, we depict the Elliptic Curve Encryption Scheme and the Elliptic Curve Digital Signature Algorithm. Our proposed mechanism is shown in Section 3. Next, we analyze the possible attacks on this mechanism in Section 4. We discuss the requirements met and the comparisons with other mechanisms in Section 5. Finally, the conclusions are drawn in Section 6.

2 Preliminaries

In this section, we briefly describe the Elliptic Curve Encryption Scheme [14] and the Elliptic Curve Digital Signature Algorithm (ECDSA) [3]. The details of the elliptic curve system are as follows:

Let p be a prime number. Assume an elliptic curve E_p is defined by an equation:

$$E_p : y^2 = x^3 + ax + b (\mathrm{mod}\ p),$$

where $a, b \in Z_p^*$ satisfies the equation $4a^3 + 27b^2 \neq 0 (\mathrm{mod}\ p)$. For all points $P, Q \in E_p$, the rules for addition over E_p are defined as:

1. $P + O = O + P = P$, where O denotes infinity. If we can find $n \bullet P = O$, n is the order of the point P. In the elliptic curve cryptosystem, we can find a suitable $n > 2^{160}$.
2. If $P = (x_p, y_p)$, then $P + (-P) = (x_p, y_p) + (x_p, -y_p) = O$. The point $(x_p, -y_p)$ is the negative of P, denoted as $-P$.
3. If $P = (x_p, y_p)$ and $Q = (x_q, y_q)$ with $P \neq -Q$, then $R = P + Q = (x_r, y_r)$ and is determined by the following rules:

$$x_r = (\lambda^2 - x_p - x_q) \bmod p,$$

$$y_r = (\lambda(x_p - x_r) - y_p) \bmod p,$$

where λ is given as:

$$\text{If } P = Q,\ \lambda = \left(\frac{3x_p^2 + a}{2y_p}\right) \bmod p.$$

$$\text{If } P \neq Q,\ \lambda = \left(\frac{y_q - y_p}{x_q - x_p}\right) \bmod p.$$

2.1 Elliptic curve encryption scheme

This section introduces how to encrypt the plaintext m using the elliptic curve cryptosystem. Assume Alice wants to send an encrypted message to Bob. To use the elliptic curve system, we must know key_A is Alice's private key and $Key_A = key_A P$ is Alice's public key, where P is a base point over E_p and n is the order of the point P. Similarly, we know key_B is Bob's private key and $Key_B = key_B P$ is Bob's public key.

Step 1: Alice first converts the plaintext m to an elliptic curve point $M \in E_p$.

Step 2: Then Alice selects a random number $k \in Z_n^*$.

Step 3: Next, Alice computes the ciphertext $C = (kP, M + kKey_B) = (X_1, Y_1)$, where X_1, Y_1 are the elliptic curve points, respectively. Alice sends C to Bob.

Step 4: When Bob receives the ciphertext C, Bob can reveal M by computing the equation

$$M = Y_1 - key_B X_1 .$$

Step 5: Subsequently, Bob converts M back to m.

Using these steps, we can encrypt the message using the elliptic curve cryptosystem.

2.2 Elliptic curve digital signature algorithm (ECDSA)

Here, we show how to generate the signature using the elliptic curve cryptosystem. First, the signer chooses G is the base point over E_P, and n as the order of the point G. key_S is the signer's private key, and $Key_S = key_S G$ is the signer's public key.

Step 1: The signer selects a random number k, where $n-1 \geq k \geq 1$.

Step 2: Then the signer computes $kG = (x_1, y_1)$ and $r = x_1 \bmod n$. If $r = 0$, then go back to Step 1.

Step 3: The signer computes $s = k^{-1}\{h(m) + key_S r\} \bmod n$, where m is the message we want to be signed and h is a one-way hash function. If $s = 0$, go back to Step 1.

After these steps, we can compute the signature (r, s). To verify the signature, several steps are followed:

Step 1: After receiving the signature, the verifier computes $w = s^{-1} \bmod n$.

Step 2: The verifier computes $u_1 = h(m)w \bmod n$ and $u_2 = rw \bmod n$.

Step 3: The verifier computes $u_1 \bullet G + u_2 \bullet Key_S = (x_0, y_0)$ and $v = x_0 \bmod n$.

Step 4: The verifier determines whether $v \stackrel{?}{=} r$ holds or not. If it holds, the signature is correct.

3. Proposed mechanism

In this section, we present a novel e-check mechanism using an elliptic curve cryptosystem. There are three roles in our mechanism—Customer, Merchant, and Bank—and two phases, the Initialization Phase and the Paying Phase. The notations used in the new mechanism are shown in Table 1, and the flowchart of the new mechanism is illustrated in Fig.1 and Fig.2.

Table 1. The notations used in the new mechanism

Notations	Description of notations
$h_1(x)$	the one-way hash function used to hash the value x
$h_2(X)$	The one-way hash function used to hash the point X over the elliptic curve E_p
c, m, b	the private key of Customer, Merchant, and Bank, respectively
$C_{pub}, M_{pub}, B_{pub}$	the public key of Customer, Merchant, and Bank, respectively
(X, Y)	X and Y are the points over an elliptic curve E_p, respectively
(x, y)	x and y are the x-coordinate and y-coordinate values of a point over an elliptic curve E_p
C_Z	the ciphertext generated by a role Z

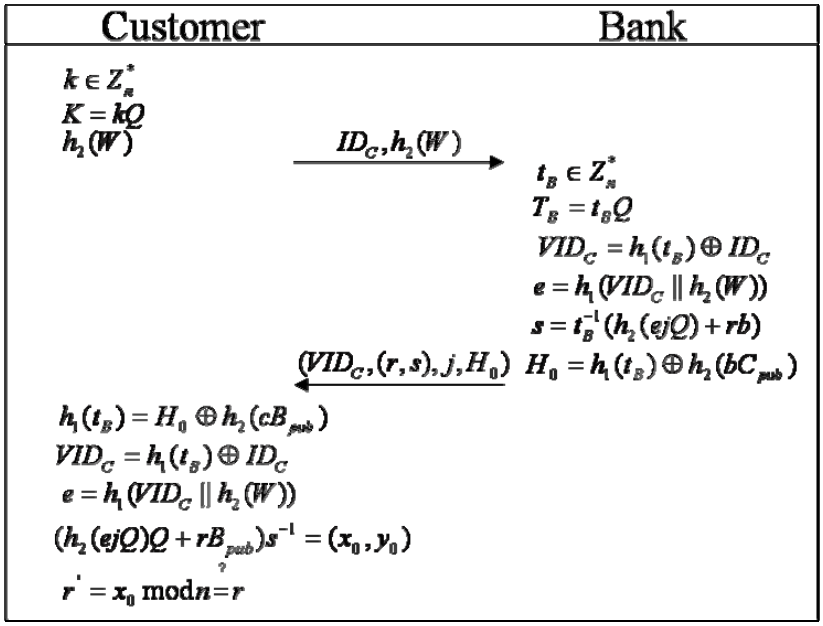

Fig. 1. The initialization phase

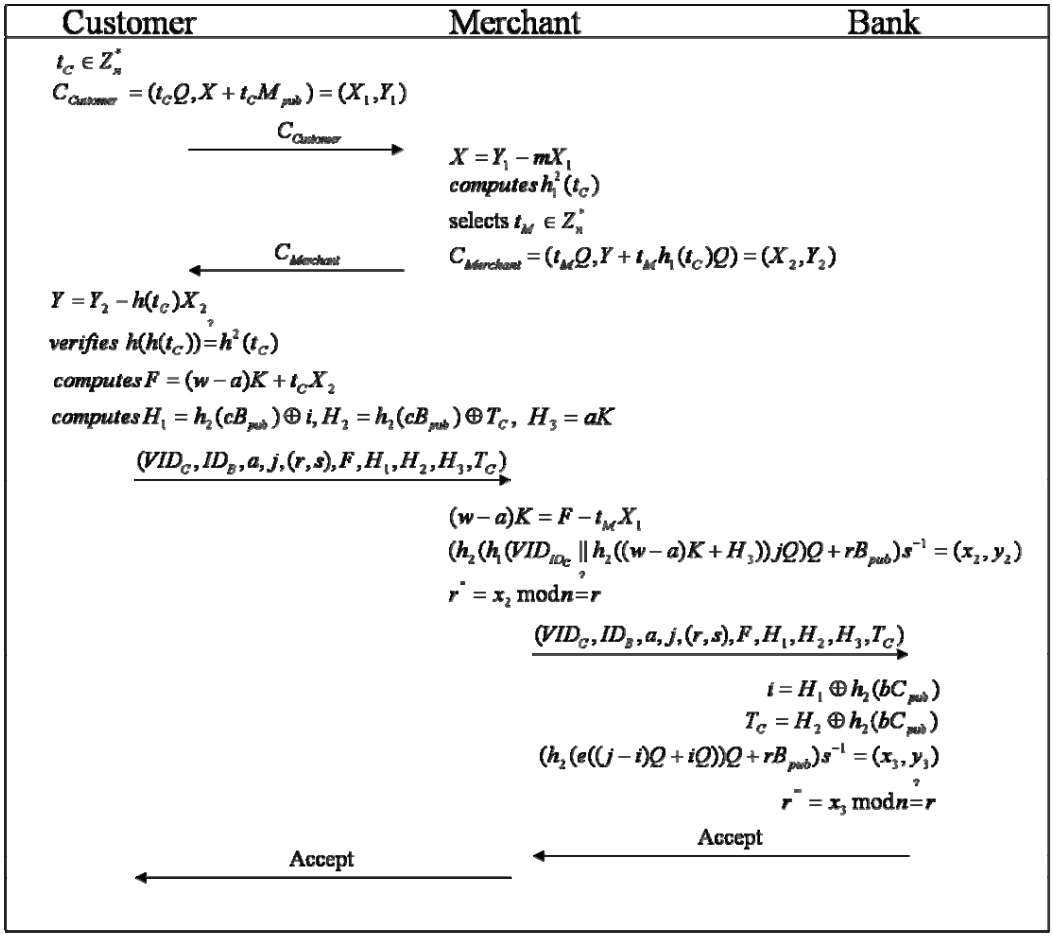

Fig. 2. The paying phase

3.1 Initialization phase

In this phase, the Customer and Merchant create an account with the Bank. Let p be a large prime number in the elliptic curve cryptosystem. The Customer selects $c \in Z_n^*$ to be the private key and computes $C_{pub} = cQ$ to be the public key, where Q is the base point over an elliptic curve E_p and n is the order of the point Q. The Bank and the Merchant use the same method to obtain their private keys and public keys. (b, B_{pub}) is the key pair of the Bank and (m, M_{pub}) is the key pair of the Merchant. Assume w is the maximum face value of the e-check.

Step 1: The Customer selects a secret number $k \in Z_n^*$ and computes

$$K = kQ .$$

and,

$$h_2(W) = h_2(wK) .$$

Then the Customer sends the message $(ID_C, h_2(W))$ to the Bank, where ID_C is the real identity of the Customer.

Step 2: Upon receiving the message, the Bank also selects a random number $t_B \in Z_n^*$ and computes

$$T_B = t_B Q = (x_1, y_1)$$

and

$$VID_C = h_1(t_B) \oplus ID_C ,$$

where VID_C is the virtual identity of the Customer. In addition, the Bank computes $r = x_1 \bmod n$, where x_1 is the x-coordinate value of T_B.

Step 3: The Bank computes

$$e = h_1(VID_C \| h_2(W)) ,$$
$$s = t_B^{-1}(h_2(ejQ) + rb) ,$$

and

$$H_0 = h_1(t_B) \oplus h_2(bC_{pub}) ,$$

where j is the maximum times the Customer can use the e-checkbook. Then the Bank sends $(VID_C, (r,s), j, H_0)$ to the Customer, where (r,s) is a signature.

Step 4: After the Customer receives the message, the Customer obtains $h_1(t_B)$ by computing $h_1(t_B) = H_0 \oplus h_2(cB_{pub})$. Next, the Customer uses $h_1(t_B)$ to compute

$$VID_C = h_1(t_B) \oplus ID_C$$

and

$$e = h_1(VID_C \| h_2(W)) .$$

Step 5: Next, the Customer verifies the signature (r,s) by computing the follow equations

$$[h_2(ejQ)Q + rB_{pub}]s^{-1} = (x_0, y_0) ,$$

and

$$r' = x_0 \bmod n \stackrel{?}{=} r .$$

If the equations are valid, the Customer stores the $VID_C, e, (r,s)$ and j in the e-check book.

3.2 Paying phase

In this phase, the Customer completes the transaction for purchasing goods with an e-check. Assume that a is the face value of this transaction, where $a \le w$.

Step 1: The Customer selects a random number $t_C \in Z_n^*$ and converts $h_1(t_C)$ to a point X over an elliptic curve E_p. Then the Customer encrypts X by computing

$$C_{Customer} = (t_C Q, X + t_C M_{pub}) = (X_1, Y_1),$$

where $C_{Customer}$ is the ciphertext generated by VID_C. Then the customer sends $C_{Customer}$ to the Merchant.

Step 2: After receiving the message, the Merchant retrieves X by computing

$$X = Y_1 - mX_1.$$

Then the Merchant converts X back to $h_1(t_C)$ and computes $h_1^2(t_C)$.

Step 3: Next, the Merchant converts $h_1^2(t_C)$ to a point Y over an elliptic curve E_p and encrypts Y by computing

$$C_{Merchant} = (t_M Q, (Y + t_M h_1(t_C) Q)) = (X_2, Y_2),$$

where $C_{Merchant}$ is the ciphertext generated by ID_M, and t_M is a random number selected by the Merchant. Then the Merchant sends $C_{Merchant}$ to the Customer.

Step 4: When the Customer receives the message, he obtains Y by computing

$$Y = Y_2 - h_1(t_C) X_2.$$

Then, the Customer converts Y back to $h_1^2(t_C)$ and checks its validity by the equation

$$h_1(h_1(t_C)) \stackrel{?}{=} h_1^2(t_C).$$

If the validity holds, the Customer performs Step 5; otherwise, the Customer terminates.

Step 5: The Customer computes

$$F = (w - a)K + t_C X_2,$$

$$H_1 = h_2(cB_{pub}) \oplus i,$$

and

$$H_2 = h_2(cB_{pub}) \oplus T_C,$$

where a is the face value of the transaction and $a \le w$, i is the number of times the e-checkbook has been used, and T_C is the timestamp generated by the Customer. The use of a timestamp, which can resist a reply attack, requires a synchronization mechanism among the Customer, the Merchant and the Bank.

Step 6: The Customer uses the secure number k to compute

$$H_3 = aK = akQ.$$

Then the Customer sends $(VID_C, ID_B, a, j, (r,s), F, H_1, H_2, H_3, T_C)$ to the Merchant, where ID_B is the identity of the Bank, j is the maximum times the Customer can use the e-checkbook, and (r,s) is the signature signed by the Bank.

Step 7: After receiving the Customer's message, the Merchant retrieves $(w-a)K$ from F by computing

$$(w - a)K = F - t_M X_1$$

and then uses $(w-a)K$ and H_3 to compute

$$(h_2(h_1(VID_C \,\|\, h_2((w-a)K + H_3))jQ)Q + rB_{pub})s^{-1} = (x_2, y_2).$$

Next, the Merchant verifies whether the equation

$$r^{*} = x_2 \bmod n \overset{?}{=} r$$

holds. If the equation holds, the Merchant sends $(VID_C, ID_B, a, j, (r,s), F, H_1, H_2, H_3, T_C)$ to the Bank.

Step 8: Upon receiving the message, the Bank inquiries into the Customer's deposit and the real identity of ID_C by VID_C. With the ID_C, the Bank can check whether the Customer's deposit is sufficient for the transaction. If it is not, the transaction is terminated; otherwise, the Bank continues the transaction.

Step 9: The Bank obtains i and T_C by computing

$$i = H_1 \oplus h_2(bC_{pub})$$

and

$$T_C = H_2 \oplus h_2(bC_{pub}) .$$

Then the Bank verifies whether the timestamp T_C is within a legal time interval. If it is not, the e-check is rejected; otherwise, the procedure continues.

Step 10: The Bank computes

$$(h_2(e((j-i)Q + iQ))Q + rB_{pub})s^{-1} = (x_3, y_3)$$

$$r^{''} = x_3 \bmod n \overset{?}{=} r$$

to determine whether the equation holds. If it holds, the Bank updates the Customer's e-checkbook, deducts *a* dollars from the Customer's account, adds *a* dollars to the Merchant's account, and sends an "Accept" message to the Merchant; otherwise, the Bank terminates the procedure.

Step 11: After receiving the "Accept" message from the Bank, the Merchant sends an "Accept" message to the Customer; otherwise, the Merchant abandons the procedure. After this step, the transaction is complete.

4 Security analysis

In this section, we analyze the security requirements of our proposed mechanism. The proposed mechanism can withstand double spending, the replay attack, the forgery attack, the impersonation attack, the DoS attack, and e-checkbook loss. First, however, we address briefly two mathematical problems with the elliptic curve E_p.

1. Elliptic Curve Diffie-Hellman problem (ECDLP): Given two points $P, Q \in E_p$, it is difficult to find an integer $a \in Z_n^{*}$, such that $Q = aP$.

2. Computational Diffe-Hellman problem (CDHP): Given aP and bP, where $a, b \in Z_n^{*}$, it is difficult to compute abP.

4.1 Double spending

Assume that the Merchant forwards the already-used message $(VID_C, ID_B, a, j, (r,s), F, H_1, H_2, H_3, T_C)$ to the bank to initiate double spending. After the Bank receives the already-used message, the Bank will inquire whether this message is already in the database. If the same record is in the database, the Bank will reject the transaction. Thus, our proposed mechanism can prevent double spending.

4.2 Replay attack

Assume that Eve is a malicious attacker and she wants to replay a message to the Bank that she has intercepted. First, Eve sends the message $(VID_C, ID_B, a, j, (r,s), F, H_1, H_2, H_3, T_C)$ to the Bank. After receiving the message, Bank checks to see whether the timestamp T_C is within a legal time interval or not. If it isn't, the Bank terminates this transaction, foiling the replay attack.

If, however, Eve generates a new timestamp T_E instead of T_C, where T_E is a current timestamp, the new timestamp T_E will be in a legal time interval and pass the authentication step when the Bank receives the replay message $(VID_C, ID_B, a, j, (r,s), F, H_1, H_2, H_3, T_E)$. However, Eve still fails in Step 9 of the Payment Phase because the Bank will decrypt H_2 to obtain a timestamp T_C and check to see whether it equals the new timestamp T_E. Since it will not, the Bank will terminate the transaction, again foiling the replay attack.

4.3 Forgery attack

Assume that the attacker Eve wants to send the forged message

$$(VID_C, ID_B, a, j, (r', s'), F', H_1', H_2', H_3', T_C')$$

to the Merchant to pass through the verification, where $\{(r', s'), F', H_1', H_2', H_3', T_C'\}$ means the forged message. After receiving the forged message, the Merchant computes the equation

$$h_2(\alpha jQ) + rB_{pub}]s^{-1} = (x_2, y_2)$$

and verifies

$$r' = x_2 \bmod n \stackrel{?}{=} r$$

in Step 7 of the Payment Phase.

In order to produce the forged message F', Eve has to know the two random numbers t_C and t_M so Eve will fail to pass the authentication equation. Even if we assume that Eve can intercept the messages $X_1 = (t_C Q)$ and $X_2 = (t_M Q)$, it is still impossible for her to obtain the correct $t_C t_M Q$ because the two random numbers are based on the difficulty of computing CDHP. In addition, Eve must know the random number k from $K = kQ$ in order to produce the forged message H_3'; since, the point *K* never appears in the transmitted message, Eve cannot produce the correct message to pass the authentication equation. Even if Eve can get the *K*, it is still impossible for her to derive k from *K* because the difficulty of computing CDHP

Similarly, if Eve forwards the forged message to the Bank, she must fail; because of the difficulty of computing CDHP, Eve cannot obtain the correct cbQ from $C_{pub} = cQ$ and $B_{pub} = bQ$. Without knowing the correct cbQ, it is impossible for Eve to forge the messages H_1' and H_2' to pass the authentication equation.

4.4 Impersonation attack

In this case, the malicious attacker Eve wants to impersonate the Customer in order to use the Customer's e-checkbook. First, Eve communicates with the Merchant to get the message $t_C'' t_M'' Q$. Then Eve wants to use $t_C'' t_M'' Q$ to produce a valid

$$F'' = (w-a)K'' + t_C'' t_M'' Q$$

and sends the message $(VID_C, ID_B, a, j, (r,s), F'', H_1'', H_2'', H_3'', T_C)$ to the Merchant, where $K'' = k''Q$. The Merchant retrieves $(w-a)K''$ from F'' and then computes

$$[h_2(h_1(VID_C \,\|\, h_2((w-a)K'' + H_3''))jQ) + rB_{pub}]s^{-1} = (x_2'', y_2'').$$

Then the Merchant verifies

$$r'' = x_2'' \bmod n \stackrel{?}{=} r.$$

However, it is impossible to compute the F'' that can pass the verification because Eve cannot compute the correct $K = kQ$ without knowing the secret number k. Hence, our mechanism can resist the imper-

sonation attack.

4.5 DoS attack

Assume that a legal Customer Eve wants to perform a DoS attack on the proposed mechanism. When the Bank receives the request message, the Bank retrieves the used time *i* from H_1. Then the Bank computes

$$(h_2(e((j-i)Q+iQ))Q+rB_{pub})s^{-1}=(x_3,y_3)$$

to determine whether the equation

$$r'' = x_3 \bmod n \stackrel{?}{=} r$$

holds. If it holds, the Bank updates the Customer's e-checkbook but, in the Initialization Phase, the Bank limited the number of times the e-checkbook can be used, so our mechanism can withstand the DoS attack.

4.6 E-checkbook loss

Assume that the attacker Eve can steal an e-checkbook from someone and that Eve wants to impersonate the Customer in order to use this e-checkbook. First, Eve communicates with the Merchant to compute

$$t_C^{\#}t_M^{\#}Q$$

and

$$F^{\#}=(w-a)K^{\#}+t_C^{\#}t_M^{\#}Q\ ,$$

where $K^{\#}=k^{\#}Q$. Then Eve sends $(VID_C, ID_B, a, j, (r,s), F^{\#}, H_1^{\#}, H_2^{\#}, H_3^{\#}, T_C)$ to the Merchant. The Merchant retrieves $(w-a)K^{\#}$ from $F^{\#}$ and computes

$$(h_2(h_1(VID_C \,\|\, h_2((w-a)K^{\#}+H_3^{\#}))jQ)Q+rB_{pub})s^{-1}=(x_2^{\#},y_2^{\#})$$

to verify that

$$r^{\#} = x_2^{\#} \bmod n \stackrel{?}{=} r\ .$$

However, Eve will not pass these authentication equations because, without knowing the random number *k*, Eve cannot compute the message $F^{\#}$ is equal to F. Therefore, Eve cannot use a legal e-checkbook that belongs to someone else.

5 Conclusions

This section addresses how our proposed scheme can achieve the requirements of the e-check: user privacy, mutual authentication, uniqueness, robustness, and non-repudiation. We also compare other methods to ours.

5.1 User privacy

In the Initialization Phase, we use the virtual identity VID_C to replay with the Customer's real identity ID_C. The VID_C is generated by computing $VID_C = h_1(t_B) \oplus ID_C$. In our mechanism, only the Bank and the Customer can obtain $h_1(t_B)$, so it is impossible for the Merchant to retrieve any information about the Customer from VID_C. However, when a valid transaction occurs, the Bank can retrieve the real identity ID_C from VID_C and use it to find the verifiable information in the database. Thus, our mechanism provides the user with privacy.

5.2 Mutual authentication

In Step 4 of the Payment Phase, the Customer decrypts the message $C_{Merchant}$ to reveal $h_1^2(t_C)$. Then the Customer authenticates the Merchant by the equation

$$h_1(h_1(t_C)) \stackrel{?}{=} h_1^2(t_C)$$

because only a valid Merchant can reveal the $h_1(t_C)$ from $C_{Customer}$ and compute $h_1^2(t_C)$. In addition, t_C is protected under the hash function, where t_C is a random number selected by the Customer. Therefore, if the equation holds, the Customer can authenticate that the Merchant is valid.

On the other hand, when the Merchant receives the message from the Customer in Step 7, the Merchant verifies whether the Customer is valid by computing

$$(h_2(h_1(VID_C \| h_2((w-a)K + H_3))jQ)Q + rB_{pub})s^{-1} = (x_2, y_2)$$

and

$$r^* = x_2 \bmod n \stackrel{?}{=} r$$

because only the valid Merchant can compute the correct $(w-a)K$ by the equation $(w-a)K = F - t_M X_1$. No one can compute $K = kQ$ without knowing the secret number *k*; therefore, if the equation holds, the Merchant can authenticate that the Customer is valid.

As a result, our mechanism can meet the requirement for mutual authentication.

5.3 Uniqueness

In the Initialization Phase, each Customer uses the ID_C and $h_1(t_B)$ to get the corresponding virtual identity VID_C. According to the VID_C, $h_2(W)$, and hash function h_1, the Bank and the Customer can cooperate to generate an e-checkbook. Because each VID_C corresponds to one unique identity ID_C, it is impossible to generate the same e-check book twice. As the result, our proposed mechanism satisfies the requirement of uniqueness.

5.4 Robustness

In the Initialization Phase, the information $e = h_1(VID_C \| h_2(W))$ for the e-checkbook is protected under the hash functions, h_1 and h_2, and $W = wK = wkQ$. Without knowing the secret number *k*, it is infeasible to compute *W*. Therefore, no one except the Customer and the Bank, who know the secret number, can generate the information e because of the security of hash function and ECDLP. Consequently, our proposed scheme meets the requirement of robustness.

5.5 Non-repudiation

To guarantee fairness to participants in the e-check mechanism, the mechanism must be able to prevent the Customer from denying an e-check that he has used in a transaction with the Bank. If the Customer wants to deny the transaction, the Bank can detect this by verifying the common session key cbQ because only the legal Customer can compute the correct $cB_{pub} = cbQ$. Thus, our proposed scheme meets the requirement of non-repudiation.

5.6 Comparisons

Here, we compare our method's computation load, the ability to prevent attack, and ability to meet the requirements of e-check with those of Pasuqathinathan, Pieprzyk and Wang [15] and Chen [1]. Table 2 and Table 3 compare the ability to prevent attacks and satisfy requirements and show that our proposed mechanism can prevent attacks and meet all the requirements of e-check mechanisms. In Table 2, **Y**

means that the attack can be prevented, and **N** means that the attack cannot be prevented. In the same way, in Table 3, **Y** and **N** mean that the requirement can be achieved and the requirement cannot be achieved, respectively.

Table 2. The attacks comparisons

	Our scheme	Chen's scheme [1]	Pasupathinathan et al.[15]
Double Spending	Y	Y	Y
Replay attack	Y	N	Y
Forgery attack	Y	Y	Y
Impersonation attack	Y	Y	N
DoS attack	Y	N	N
e-check book loss	Y	Y	Y

Table 3. The requirements comparisons

	Our scheme	Chen's scheme [1]	Pasupathinathan et al.[15]
User Privacy	Y	N	Y
Mutual Authentication	Y	N	N
Uniqueness	Y	N	Y
Robustness	Y	Y	Y
Non-Repudiation	Y	N	Y

Comparing the approximate value between the elliptic curve and modular exponentiation in Lee, Wu and Wang [16], the time for modular exponentiation T_Exp is approximately 240 modular multiplication T_Mul. The time of multiplication and addition over an elliptic curve are approximately 29 and 5 modular multiplication T_Mul, respectively.

We compare our mechanism with other related mechanisms in Table 4 to show that our proposed mechanism needs approximately $722T\_Mul$ computation time, while the other two methods need approximately $3840T\_Mul$ and $960T\_Mul$ computation time, respectively. The computation times of the hash function and the XOR operation are much lower than the time of modular multiplication, so we do not consider them in Table 4.

Table 4. The efficiency comparisons

	Initialization Phase	Paying Phase	Total T_Mul
Our scheme	$9T\_Mul_{EC}$ $1T\_Add_{EC}$	$14T\_Mul_{EC}$ $10T\_Add_{EC}$	$722T\_Mul$
Chen's scheme [1]	$2T\_Exp$	$2T\_Exp$	$960T\_Mul$
Pasupathinathan et al.[15]	$5T\_Exp$	$11T\_Exp$	$3840T\_Mul$

6 Conclusions

We proposed an e-check protocol based on an elliptic curve, analyzed the method's ability to prevent attacks, discussed how it meets the requirements of e-checks, compared its computation time with that of other methods, and found that our proposed mechanism is more secure and more efficient than other e-check mechanisms.

References

[1] Chen, W. K. (2005). Efficient on-line electronic checks. *Applied Mathematics and Computing, 162*(3), 1259-1263.

[2] Chaum, D., Fiat, A., & Naor, M. (1990, August). *Untraceable electronic cash*. Paper presented at the Proceedings of Advances in Crypto'88, LNCS 403, Santa Barbara, CA.

[3] Hsin, W. J., & Harn, L. (2005, March). *Simple certified e-check with a partial PKI solution*. Paper presented at the Proceedings of the 43rd Annual Southeast Regional Conference, Kennesaw, GA.

[4] Yu, H. C., His, K. H., & Kuo, P. J. (2002). Electronic payment systems: An analysis and comparison of types. *Technology in Society, 24*(3), 331-347.

[5] Rui, X. (2010, August). *Secure e-check payment model based on ECC*. Paper presented at the Proceedings of 2010 WASE International Conference on Information Engineering, BeiDai, China.

[6] Xu, Y., & Liu, J. (2009, September). *Electronic check system design based on NFC*. Paper presented at the Proceedings of International Conference on Management and Service Science, Beijing, China.

[7] Chaum, D., Den Boer, B., Van Heyst, E., Mjolsnes, S., & Steenbeek, A. (1989, April). *Efficient offline electronic check*. Paper presented at the Proceedings of Advances in Eurocrypt'89, Berlin, Germany.

[8] Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM, 21*(2), 120-126.

[9] ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory, 31*, 465-472.

[10] Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of Computation, 48*, 203-209.

[11] Miller, V. (1986, ?). *Use of elliptic curves in cryptography*. Paper presented at the Proceedings of Advance in Crypto'85, LNCS, Berlin, Germany.

[12] Diffie, W., & Hellman, M. (1976). New directions in cryptography. *IEEE Transactions on Information Theory, 22*(6), 644-654.

[13] Chang, C. C., Chang, S. C., & Lee, J. S. (2009). An on-line electronic check system with mutual authentication. *Computers & Electrical Engineering, 35*(5), 757-763.

[14] Johnson, D., & Menezes, A. (1999). *The elliptic curve digital signature algorithm (ECDSA), Technical Report CORR 99-34.* Retrieved from http://www.cacr.math.uwaterloo

[15] Pasupathinathan, V., Pieprzyk, J., & Wang, H. (2005, July). *Privacy enhanced electronic cheque system*. Proceedings of the Seventh IEEE International Conference on E-Commerce Technology, Munich, Germany.

[16] Lee, N. Y., Wu, C. N., & Wang, C. C. (2008). Authenticated multiple key exchange protocols based on elliptic curves and bilinear pairings. *Computers and Electrical Engineering, 34*(1), 12-20.