# Cryptanalysis of Some Electronic Checkbook Schemes

Isa Sertkaya[a] and Oznur Kalkar[b]
*MCS Labs & BCLabs, TÜBİTAK BİLGEM UEKAE, PK. 74, 41470, Gebze, Kocaeli, Turkey*

Keywords: Electronic Checkbook, e-Checkbook, e-Check, Cryptanalysis, Security, Privacy.

Abstract: Paper-based check is the second mostly used payment method. Accordingly, efforts are underway to improve electronic checkbook (shortly, e-checkbook) systems which mimics the paper-based checkbook mechanism, in line with social needs. Considering the cost of paper check procedures and the amount of money transferred using checks, we believe that there should be a properly designed and provably secure e-checkbook scheme. Analyzing the vulnerabilities of the existing systems, and figuring out where they originate is the first step towards a secure e-checkbook mechanism. In this study, we show that the e-checkbook schemes denoted as PEEC, CYLL, CCL, CWL and CCW fail to achieve their claimed security and susceptible to various types of attacks including e-check forgery and manipulation. Particularly, we show that Pasupathinathan *et al.*'s PEEC scheme does not satisfy the correctness, anonymous identity and payment unlinkability; Chen *et al.*'s CYLL scheme is not secure against e-check manipulation and e-check forgery attacks; Chang *et al.*'s CCL scheme, Chen *et al.*'s CWL scheme and Chang *et al.*'s CCW scheme are susceptible to e-check manipulation attack.

## 1 INTRODUCTION

Check-based payments constitute around 10 percent of payments which corresponds to 16 billion checks transferring 27 trillion USD in USA in 2016. Since these checks are needed to be processed manually, they require considerable amount of corporate resources. For example, the average cost for a corporation to process a check is US$1.5 as stated by Chief Executive of Hong Kong Monetary Authority, (Chan, 2015). As a direct conclusion, in 2016, processing the 16 billion checks in USA would cost around 24 billion dollars. This 24 billion dollars, or a large portion of it could be saved if e-checkbook was used.

A paper check is a payment form that draws money from a checking account when deposited. The *payer* first gathers a checkbook which is a collection of empty checks issued by the payer's bank. Whenever a *payer* wants to pay with a paper check, he/she writes the name of the *payee* and the desired *face value* (amount) on the check and signs it. After receiving a check, the payee presents the received check to the bank and requests the deposit.

An electronic check, e-check for short, on the other hand, is electronic version of a paper check. Whenever a payer wants to make a payment using e-check, she writes the payee and the face value information to the check and digitally signs it. Early e-check proposals (Chaum et al., 1990b; Chaum et al., 1990a; Brands, 1993; Chen, 2005) require e-check issuance before each payment which is a drawback, since it requires the payer to run a protocol jointly with the bank before each payment. However, e-checks can be bundled into an e-checkbook as in the paper check system.

Up to our knowledge, there are seven e-checkbook schemes (Pasupathinathan et al., 2005; Chen et al., 2009; Chang et al., 2009; Chen et al., 2010; Chang et al., 2016; Sertkaya and Kalkar, 2019; Sertkaya and Kalkar, 2021), that do not require e-check issuance before each payment. As we stated earlier, considering the cost of paper check clearing / settlement processes and the volume of money transferred by checks, there is a need for a properly designed and provably secure e-checkbook scheme. Analyzing the vulnerabilities of the existing systems and figuring out where they originate is the first step towards a secure e-checkbook mechanism.

***Related Work.*** Chaum (Chaum et al., 1990b) introduced the idea of electronic check and proposed an offline e-check system. Some examples of other e-check mechanisms are (Chaum et al., 1990a; Brands, 1993; Chen, 2005; Katz and Lindell, 2014). However, all of the aforementioned propositions needs the

[a] https://orcid.org/0000-0002-4739-0515
[b] https://orcid.org/0000-0002-7875-3892



Sertkaya, I. and Kalkar, O.
Cryptanalysis of Some Electronic Checkbook Schemes.
DOI: 10.5220/0010981900003120
In *Proceedings of the 8th International Conference on Information Systems Security and Privacy (ICISSP 2022)*, pages 603-612
ISBN: 978-989-758-553-1; ISSN: 2184-4356

payer to interact with the issuer bank for each and every check issuance, hence they do not fully simulate paper-based checkbook system. In this paper, we mainly consider the e-checkbook solutions and focus on the mechanisms that simulate the paper-based checkbook system.

In 2005, following FSTC e-check scheme (Anderson, 1998) and Check 21 Act (Check 21 Act, 2003), Pasupathinathan et al. highlighted privacy issues in e-check schemes, and proposed the first e-checkbook scheme PEEC, (Pasupathinathan et al., 2005). In PEEC, at the end of the issuing phase, the payer is given different Schnorr signatures (Katz and Lindell, 2014) for each e-check by the issuing bank. Following the e-check mechanism given in (Chen, 2005), three e-checkbook schemes are proposed. First, (Chen et al., 2009) modified the scheme into an e-checkbook system where the e-checkbook can be issued with only one signature of the issuer bank. Next, (Chang et al., 2009) proposed another version that enables e-checkbook issuance and mutually authenticated payment, but introduced time synchronization issues. Lastly, (Chen et al., 2010) claims to improve (Chang et al., 2009) computationally at the cost of increasing number of protocol rounds involving payer, payee and the issuer bank. (Chang et al., 2016) gives another e-checkbook mechanism based on elliptic curve cryptography. In the pursue of designing secure e-checkbook scheme, the authors proposed two e-checkbook schemes; (Sertkaya and Kalkar, 2019) that satisfies mutual authentication of the payer and the payee and more recently (Sertkaya and Kalkar, 2021) that supports transferable e-checks and satisfies anonymity property against eavesdropper.

***Our Contributions.*** In this study, we focus on the security analysis of the previously proposed e-checkbook schemes except (Sertkaya and Kalkar, 2019; Sertkaya and Kalkar, 2021).

More concretely, we show that PEEC does not satisfy the correctness, anonymous identity and payment unlinkability, CYLL is not secure against e-check manipulation and e-check forgery attacks, CCL is susceptible to e-check manipulation attack, CWL is vulnerable against e-check manipulation attack, CCW is susceptible to e-check manipulation attack.

***Organization.*** In Section 2, we define the e-checkbook architecture and known attack types. We analyze (Pasupathinathan et al., 2005), (Chen et al., 2009), (Chang et al., 2009), (Chen et al., 2010), and (Chang et al., 2016) in Sections 3, 4, 5, 6, and 7, respectively. Finally, we discuss additional privacy concerns and conclude the manuscript with Section 8.

# 2 DEFINITIONS AND SECURITY NOTIONS

These entities involved in an e-checkbook have the same roles as in the paper-check solutions.

- *Payer* is an entity who has a registered account with the issuer bank, wants to get an e-checkbook, and use e-checks to make payments to another entity.
- *Payee* is an entity who received an e-check from a Payer, wants to deposit this e-check to the acquirer bank and waits for the corresponding money transaction to be finalized.
- *Issuer* is the bank of the Payer who issues the e-checkbook to its registered users. In case of an e-check presentment Issuer also initiates the actual money transfer from the Payer's account to the Payee's account.
- *Acquirer* is the bank who holds the Payee's registered account and whenever a Payee presents an e-check, Acquirer initiates inter-bank transactions to finalize the actual money transfer.

For the sake of simplicity, we assume that the Issuer and the Acquirer banks are the same. We denote the bank by B, the Payer by U, and the Payee by M.

An e-checkbook scheme consists of four phases; namely Initializing, Issuing, Paying, and Depositing phases.

- Initializing. Given a security parameter, system environment, public parameters, private and public key pair for each entity are generated.
- Issuing. User U and the bank B create a valid e-checkbook for U.
- Paying. Upon agreeing on the date and amount with M, the payer U creates an e-check and sends it to M.
- Depositing. Whenever a payee M receives an e-check payment, she verifies its authenticity and forwards to the bank B. After completing the necessary controls, B deducts the amount from U's account, transfers it to M's account and informs M.

## 2.1 Attack Types

***e-checkbook Forgery.*** A malicious entity collects e-checkbooks issued by B for different users and creates a valid e-checkbook belonging to another user as if it is issued by B.

***e-check Forgery.*** A malicious entity collects e-checks originating from U up to $i^{\text{th}}$ e-check and creates $(i+1)^{\text{th}}$ e-check belonging to U.

***e-check Manipulation.*** An adversary collects e-checks originating from U up to $i^{\text{th}}$ e-check, intercepts $(i+1)^{\text{th}}$ e-check, and manipulates $(i+1)^{\text{th}}$ e-check by combining parameters originating from multiple checks.

***Replay Attack.*** An adversary captures an e-check, and re-sends as a valid e-check.

***Double Spending Attack.*** A malicious payer spends an e-check which is already spent.

***E-check Linkability.*** Besides the payer, the payee and the bank, an adversary deduces whether the e-checks belong to a payer or a payee.

Core security requirements for an e-checkbook scheme to avoid the mentioned attacks are correctness, e-checkbook authentication, e-check authentication, e-check integrity, source authentication, and e-check anonymity.

# 3 ANALYSIS OF PEEC SCHEME

PEEC scheme, proposed in (Pasupathinathan et al., 2005), claims that it provides enhanced privacy by allowing the payer to choose an *anonymous identity* during a transaction which results in protection of the payment details like payer's account information from merchants. We are going to show that PEEC scheme does not satisfy the correctness, anonymous identity and payment unlinkability properties.

## 3.1 PEEC Protocol

Initializing

1. B chooses a subgroup $\mathbb{G}_q$ of prime order q of the multiplicative group $\mathbb{Z}_p^*$ where prime $p$ satisfies $p = \gamma q + 1$ for some specified integer $\gamma$ and selects generators $g_0, g_1, g_2$ of $\mathbb{G}_q$. B fixes a collision resistant hash function $\mathcal{H}(\cdot)$, generates her secret key $x_{\mathsf{B}} \leftarrow_{\$} \mathbb{Z}_q$ and corresponding public keys $h = g_0^{x_{\mathsf{B}}}$, $h_1 = g_1^{x_{\mathsf{B}}}$, $h_2 = g_2^{x_{\mathsf{B}}}$.[1] B publishes system-wide public parameters
$$\mathsf{pp} \leftarrow \{p, q, \mathcal{H}(\cdot), g_0, g_1, g_2, h, h_1, h_2\}.$$
2. Each payer U initially registers with B for her account indexed with $b_{\mathsf{U}} \in \mathbb{Z}_q$, generates her public key $U \leftarrow g_1^{x_{\mathsf{U}}}$ where $x_{\mathsf{U}} \leftarrow_{\$} \mathbb{G}_q$ such that $g_1^{x_{\mathsf{U}}} g_2 \neq 1$.
3. Similarly, each payee registers with B and obtains a certified public key $M \leftarrow g_1^{x_{\mathsf{M}}}$ where $x_{\mathsf{U}} \leftarrow_{\$} \mathbb{G}_q$.

Issuing

1. U sends her public key $U$ to B.
2. B randomly selects $t, k, k_1, k_2, \ldots, k_r \in_{\$} \mathbb{Z}_q$ where $r$ being the number of e-checks in the e-checkbook.
3. B computes digital token $Y \leftarrow Uy$ with $y \leftarrow g_1^t$, and signature $\sigma_Y \leftarrow Y x_{\mathsf{B}} + k \pmod q$ for $Y$.
4. Next, B generates each e-check indexed by $i$ with $1 \le i \le r$, and signs them, i.e. $\omega^i \leftarrow \mathcal{H}(U g^{b_{\mathsf{U}}} g^i)$ and $\sigma_{\omega^i} \leftarrow \omega^i x_{\mathsf{B}} + k_i \pmod n$.
5. For U's e-check post-processing, B stores $(Y, U, (\omega^i, \sigma_{\omega^i})_{1 \le i \le r}, b_{\mathsf{U}})$ in a database indexed by $Y$.
6. B sends $\{t, y, Y, \sigma_Y, (\omega^i, \sigma_{\omega^i})_{1 \le i \le r}\}$
7. U verifies each signatures separately and then stores $\Omega_{\mathsf{U}} \leftarrow (t, y, Y, \sigma_Y, (\omega^i, \sigma_{\omega^i})_{1 \le i \le r})$ as her e-checkbook.

Paying

Assume that U has already spent up to $i-1$ e-checks for some $i < r$, and is going to send $i^{\text{th}}$ e-check to M upon his signed payment request with the date $d$, face value $a$, and payee's identity M.

1. U first selects $s, w \leftarrow_{\$} \mathbb{Z}_q$, computes anonymous identity $A \leftarrow Y^s$ with $A_1 \leftarrow U^s$ and $A_2 \leftarrow y^s$.
2. Next U prepares payment by computing $o \leftarrow \mathcal{H}(d||\mathsf{M}||a)$, $v \leftarrow (x_{\mathsf{U}} s^2 t - o x_{\mathsf{U}} s)s$ and $\sigma^i_{\mathsf{UM}} \leftarrow \omega^i g^a x_{\mathsf{U}} st + w \pmod q$.
3. U sends $\omega^i_{\mathsf{UM}} \leftarrow \{v, A, A_1, A_2, o, \omega^i, \sigma_{\omega^i}, Y, \sigma_Y, \sigma^i_{\mathsf{UM}}$ to M,
4. M verifies the signatures $\sigma_{\omega^i}$ and $\sigma_Y$, and validates $o \stackrel{?}{=} \mathcal{H}(d||\mathsf{M}||a)$, $A \stackrel{?}{=} A_1 A_2$ and $A \stackrel{?}{=} A_1^o Y^v$.

Depositing

1. M chooses $z \leftarrow_{\$} \mathbb{Z}_q$, creates the signature $\sigma_o \leftarrow o x_{\mathsf{M}} + z \pmod n$ and sends the tuple $\{d, \mathsf{M}, a, v, A, A_1, A_2, o, \omega^i, Y, \sigma_Y, \sigma^i_{\mathsf{UM}}, \sigma_o\}$[2] to B.
2. B first validates $o \stackrel{?}{=} \mathcal{H}(d||\mathsf{M}||a)$ and then verifies the signatures $\sigma_Y$, $\sigma^i_{\mathsf{UM}}$ and $\sigma_o$.
3. B retrieves U's bank account number $b_{\mathsf{U}}$ and his original identity $U$, and e-check index $i$ from the database indexed by $Y$.
4. B verifies the identity of U, if she has enough funds in her account for clearance, and $i^{\text{th}}$ e-check is not already spent.

[1] PEEC scheme does not utilize $g_0$, $h$, and $h_2$, but we still included here for siding with the original proposal.

[2] PEEC authors omitted $\sigma_{\omega^i}$, but it should be included here, otherwise B has to keep records for it.

5. Finally, if the verification is successful and sufficient funds are available, B debits U's account and credits M's account.

.

### 3.2 PEEC Scheme Flaws

Even though the authors claim that they use Schnorr's signature scheme, the scheme utilizes Schnorr's identification scheme, see (Pasupathinathan et al., 2005; Katz and Lindell, 2014). In order to be able to successfully perform signature verification, each signature $\sigma$ should be sent with masked ephemeral key values, see (Katz and Lindell, 2014, Construction 12.12 on p. 458). Assuming this signature verification is fixed, the scheme still suffers from the following issues.

- *Correctness*. Even if the payer and the payee honestly follow the protocol, at the last step of Paying phase, the necessary verification of $A \stackrel{?}{=} A_1^o Y^v$ that should be pursued by the payee never holds since $g_1^{ox_U s} Y^{(x_U s^2 t - o x_U s)s} \neq Y^s$. Hence, the payee always rejects the payment.
- *Anonymous identity*. At the Issuing Phase, bank sends $t, y, Y$ together with the e-checks and the signatures to the payer. Using these values, on the contrary to the authors' claims, one can easily construct payer's public key $U$ by computing $U \leftarrow Yy^{-1}$. Hence, PEEC scheme does not satisfy anonymous identity for the payer.
- *Payment unlinkability*. At the Paying phase for each e-check, the payer always includes the digital token and bank's signature on the token $(Y, \sigma_Y)$ within the sent tuple. Hence, just by checking these values, e-check payments of the same payer will naturally be distinguished and linked.

## 4 ANALYSIS OF CYLL SCHEME

### 4.1 CYLL Protocol

The protocol assumes that B keeps user accounts as a number $b$ with $1 \leq b \leq k$. $w$ is defined as the maximal face value of an e-check, and $r$ is the number of e-checks in the e-checkbook.

Initializing

1. B chooses a one-way hash function $\mathcal{H}(\cdot)$, generates two large primes $p$ and $q$ and computes $n = pq$. B selects a public key $e$ and computes $d$ such that $e \cdot d = 1 \pmod{\phi(n)}$, keeps $x_B \leftarrow (p, q, d)$ as secret, and publishes public parameters $pp \leftarrow (n, e, \mathcal{H}(\cdot))$.

Issuing

1. U first generates four random numbers $x_1, x_2, x_3, x_4$.
2. U computes
$$m \leftarrow \mathcal{H}^w(x_1)||\mathcal{H}^w(x_2)||\mathcal{H}^k(x_3)||\mathcal{H}^k(x_4)$$
and $\alpha \leftarrow \mathcal{H}^r(m) \pmod{n}$ and sends $\{\alpha, r\}$ to B.
3. B computes $\sigma_\alpha \leftarrow \alpha^d \pmod{n}$ and sends $\sigma_\alpha$ to U.
4. U checks $\sigma_\alpha^e \pmod{n} \equiv \mathcal{H}^r(m) \pmod{n}$ holds, if so, stores $\Omega_U = (m, \sigma_\alpha, r, x_1, x_2, x_3, x_4)$ as the e-checkbook.

Paying

Assume that U has used $i-1$ $(i < r)$ e-checks and wants to attach a face value $a$ for the payee M with account number $b_M$ to the e-check for the $i^{\text{th}}$ time.

1. U computes $\beta_1 \leftarrow \mathcal{H}^a(x_1)$, $\beta_2 \leftarrow \mathcal{H}^{w-a}(x_2)$, $\beta_3 \leftarrow \mathcal{H}^{b_M}(x_3)$, $\beta_4 \leftarrow \mathcal{H}^{k-b_M}(x_4)$ and $\mathcal{H}^{r-i}(m)$.
2. U sends the e-check $\omega^i$ to M, where
$$\omega^i = (a, b_M, \sigma_\alpha, \mathcal{H}^{r-i}(m), r, i, \beta_1, \beta_2, \beta_3, \beta_4).$$

Depositing

1. M verifies the signatures, i.e,
$$\sigma_\alpha^e \pmod{n} \equiv \mathcal{H}^r(\mathcal{H}^{w-a}(\beta_1)||\mathcal{H}^a(\beta_2)|| \mathcal{H}^{k-b_M}(\beta_3)||\mathcal{H}^{b_M}(\beta_4)) \pmod{n}$$
and
$$\sigma_\alpha^e \pmod{n} \equiv \mathcal{H}^i(\mathcal{H}^{r-i}(m)) \pmod{n}.$$
2. M sends $\omega^i$ to B for double-spending check.
3. B first verifies the signatures as in Step 1.
4. B deducts the amount $a$ from U's bank account $b_U$, adds it into the M's account $b_M$, and informs M if no spend records has been found.

### 4.2 Attacks on CYLL Scheme

- *e-check Forgeability*. T.-H. Chen *et al.* claim that even if an attacker is in possession of a previously paid $i^{\text{th}}$ e-check $\omega^i$, she can not forge the next e-check and CYLL scheme is secure against "*e-check forgeability attacks*", (see (Chen et al., 2009, Section IV B.2 Unforgeability of e-check)). Unfortunately, this is not true. Based on the same assumption, we now show how an attacker A who passively intercepts this payment or a malicious payee who gets the payment can forge the next e-check by following the steps given below.

1. With the knowledge of $w, a, b_{\mathsf{M}}, k$; A computes
$$\mathcal{H}^{w-a}(\beta_1),\ \mathcal{H}^{a}(\beta_2),\ \mathcal{H}^{k-b_{\mathsf{M}}}(\beta_3),\ \mathcal{H}^{b_{\mathsf{M}}}(\beta_4).$$
2. Now, A can re-construct $m$ as
$$\begin{aligned} m &= \mathcal{H}^{w-a}(\beta_1)||\mathcal{H}^{a}(\beta_2)||\mathcal{H}^{k-b_{\mathsf{M}}}(\beta_3)||\mathcal{H}^{b_{\mathsf{M}}}(\beta_4) \\ &= \mathcal{H}^{w}(x_1)||\mathcal{H}^{w}(x_2)||\mathcal{H}^{k}(x_3)||\mathcal{H}^{k}(x_4) \end{aligned}$$
3. A computes $\mathcal{H}^{(r-(i+1))}(m)$ which is necessary for $(i+1)^{\text{th}}$ e-check.
4. A sends $(a, b_{\mathsf{M}}, \sigma_\alpha, \mathcal{H}^{(r-(i+1))}(m), r, (i+1), \beta_1, \beta_2, \beta_3, \beta_4)$, as the $(i+1)^{\text{th}}$ e-check $\omega^{i+1}$ to B, pretending it is a payment from U to M.
5. B can assure the validity of this forged e-check since the following hold:
$$\begin{aligned} \sigma_\alpha^e \pmod n &\equiv \mathcal{H}^r(\mathcal{H}^{w-a}(\beta_1)||\mathcal{H}^{a}(\beta_2)|| \\ &\qquad \mathcal{H}^{k-b_{\mathsf{M}}}(\beta_3)||\mathcal{H}^{b_{\mathsf{M}}}(\beta_4)) \pmod n, \\ \sigma_\alpha^e \pmod n &\equiv \mathcal{H}^{i+1}(\mathcal{H}^{(r-(i+1))}(m)) \pmod n. \end{aligned}$$
6. B deducts the amount $a$ from U's account $b_{\mathsf{U}}$, adds it into the M's account $b_{\mathsf{M}}$, and informs M since no records have found.

This attack shows that an attacker who has the $i^{\text{th}}$ e-check is capable of forging the $(i+1)^{\text{th}}$ e-check to the same payee with the same amount. In fact the attacker can forge all the remaining e-checks to the same payee with the same amount.

- *E-check Manipulation.* Consider that in addition to $i^{\text{th}}$ e-check, an attacker also has $j^{\text{th}}$, $j < i$, e-check with the face value greater than the $i^{\text{th}}$ e-check's face value. Then by combining these, the attacker can manipulate $(i+1)^{\text{th}}$ e-check to the payee in $i^{\text{th}}$ e-check with the face value in $j^{\text{th}}$ e-check. Hence, this scheme is also not secure against e-check manipulation.

A secure version of CYLL scheme is given by the authors in (Sertkaya and Kalkar, 2019).

# 5 ANALYSIS OF CCL SCHEME

Similar to (Chen et al., 2009), CCL scheme focuses on converting W.-K Chen's e-check scheme into an e-checkbook scheme with mutual authentication. In order to do that, this scheme uses a symmetric key cryptosystem $(E_{\mathsf{key}}(\cdot), D_{\mathsf{key}}(\cdot))$, RSA-based digital signatures, RSA-based blind signatures and timestamps, (Chang et al., 2009).

## 5.1 CCL Protocol

This scheme is originally given in only two phases, namely Registration and Paying, in order to be consistent with other sections, here we split them into Initializing, Issuing, Paying and Depositing phases.

### Initializing

1. B chooses a one-way hash function $\mathcal{H}(\cdot)$.
2. B generates two large primes $p$ and $q$ and computes $n = pq$.
3. B selects a public key $e$ and computes $d$ such that $e \cdot d = 1 \pmod{\phi(n)}$,
4. B keeps $x_{\mathsf{B}} \leftarrow (p, q, d)$ as secret, and publishes public parameters $\mathsf{pp} \leftarrow (n, e, \mathcal{H}(\cdot))$.

### Issuing

After U registers with the bank B by creating her own public-private key pair $(\mathsf{pk}_{\mathsf{U}}, \mathsf{sk}_{\mathsf{U}})$ and sharing the public key with the bank, where $\mathsf{pk}_{\mathsf{U}} = (n_{\mathsf{U}}, e_{\mathsf{U}})$ and $\mathsf{sk}_{\mathsf{U}} = (p_{\mathsf{U}}, q_{\mathsf{U}}, d_{\mathsf{U}})$, U and B follow the Issuing Phase.

1. U randomly chooses a secret integer $x_1$.
2. U computes $m = \mathcal{H}(\mathsf{U}||\mathcal{H}^w(x_1))$ and $\alpha = \mathcal{H}(m)$.
3. U sends $\{\mathsf{U}, \alpha\}$ to B.
4. B verifies the identity U. computes $\alpha' \equiv \mathcal{H}^r(m) \pmod n$, $\sigma_{\alpha'} \equiv (\alpha')^d \pmod n$ and sends $\{\sigma_{\alpha'}, r\}$ to U.
5. U verifies the integrity of the message by checking whether $(\sigma_{\alpha'})^e \pmod n \equiv \mathcal{H}^r(m) \pmod n$ or not. If it holds, U stores the e-checkbook
$$\Omega_{\mathsf{U}} = (m, \sigma_{\alpha'}, r).^3$$

### Paying

Assume that U used $i-1$ $(i < r)$ e-checks and wants to attach a face value $a$ $(a < w)$ for the payee M.

1. U randomly chooses two integers $R$ and $b$, and computes $k \equiv R^{e_{\mathsf{M}}} b \pmod{n_{\mathsf{M}}}$, where $\mathsf{pk}_{\mathsf{M}} = (e_{\mathsf{M}}, n_{\mathsf{M}})$ is the public key of the payee M.
2. U sends $k$ to M.
3. M computes $k' \equiv k^{d_{\mathsf{M}}} \equiv R b^{d_{\mathsf{M}}} \pmod{n_{\mathsf{M}}}$, where $d_{\mathsf{M}}$ is the private key of M. Then, M sends $k'$ to U.
4. U computes $M \equiv k' R^{-1} \equiv b^{d_{\mathsf{M}}} \pmod{n_{\mathsf{M}}}$, $C_1 \equiv \mathcal{H}^{w-a}(x_1) \oplus M$ and $C_2 \equiv E_{ck}(i||T)$, where $E_{ck}(\cdot)$ is a symmetric encryption with the secret key $ck$ shared between U and B, and $T$ is the current timestamp.

[3] $x_1$ should also be included in e-checkbook, since it will be needed in each Paying phase.

5. U checks if $b \equiv M^{e_M} \pmod{n_M}$ holds. If it is not valid, U terminates the transaction; otherwise, U sends the e-check $\omega^i \leftarrow (U, B, a, b, r, \sigma_{\alpha'}, C_1, C_2, T)$ to M.

Depositing

1. M verifies the integrity of $\omega^i = (U, M, a, b, r, \sigma_{\alpha'}, C_1, C_2, T)$ by checking whether
$$(\sigma_{\alpha'})^e \equiv \mathcal{H}^r(\mathcal{H}(U||\mathcal{H}^a(C_1 \oplus b^{d_M}))) \pmod{n}$$
holds. If the equation holds, M sends $\omega^i$ to B for double-spending control; otherwise, the check is rejected.
2. B checks if $\omega^i$ is already recorded in the database as spent e-check. If it is so, B rejects the e-check; otherwise, the phase continues.
3. B rejects the e-check if $a$ is greater than the payer U's deposit in the bank; otherwise, the phase continues.
4. B records the message receiving time as time-stamp $T'$. Then, B computes $i||T = D_{ck}(C_2)$, checks whether $(T' - T)$ is in acceptable valid time interval. If it is not so, the e-check is rejected.
5. B verifies the e-check by computing
$$(\sigma_{\alpha'})^e \equiv \mathcal{H}^i(\mathcal{H}^{r-i}(m)) \pmod{n}.$$[4]
If it holds, B deducts the amount $a$ from U's account $b_U$ and adds it into M's account $b_M$.

## 5.2 Flaws of CCL Protocol

Here, we first point out a few inconsistencies and then we show that CCL scheme is not resistant against e-check manipulation attack even if these inconsistencies are corrected.

- First, please note that (Chang et al., 2009) states that $(U, B, a, b, r, \sigma_{\alpha'}, C_1, C_2, T)$-tuple is transmitted as e-check, which should be in fact
$$(U, M, a, b, r, \sigma_{\alpha'}, C_1, C_2, T),$$
otherwise B can not know who the payee M is.
- Furthermore, at the Depositing phase, the authors claim that B verifies the e-check by computing
$$(\sigma_{\alpha'})^e \stackrel{?}{=} \mathcal{H}^i(\mathcal{H}^{r-i}(m)),$$
but B knows neither $m$ nor $\mathcal{H}^{r-i}(m)$. Therefore, U should compute $\mathcal{H}^{r-i}(m)$ and send the e-check
$$\omega^i \leftarrow (U, M, a, b, r, \sigma_{\alpha'}, C_1, C_2, T, \mathcal{H}^{r-i}(m))$$
to M and M should send this tuple to B.

[4] B did not receive $\mathcal{H}^{r-i}(m)$, hence can not verify this step, refer to next section.

- *E-check Manipulation*. Suppose an adversary A has
$$\omega^i = (U, M, a^i, b^i, r, \sigma_{\alpha'}, C_1^i, C_2^i, T^i, \mathcal{H}^{r-i}(m)),$$
for some $i$ with $i < r$.
1. A tracks and captures U's $(i+1)^{\text{th}}$ e-check
$$(U, M', a^{i+1}, b^{i+1}, j, \sigma_{\alpha'}, C_1^{i+1}, C_2^{i+1}, T^{i+1}, H^{j-(i+1)}(m))$$
and blocks this e-check transmission that is meant to be a payment to a different payee M'.
2. A create the manipulated e-check $\omega'$ as if it is U's $(i+1)^{\text{th}}$ e-check with the following tuple.
$$(U, M, a^i, b^i, r, \sigma_{\alpha'}, C_1^i, C_2^{i+1}, T^{i+1}, \mathcal{H}^{r-(i+1)}(m))$$
Note that $r, \sigma_{\alpha'}$ are constant for each e-check that belongs to U, $T^{i+1}, \mathcal{H}^{r-(i+1)}(m)$ are known from $\omega^{i+1}$, and $M, a^i, b^i, C_1^i$, are gathered from $\omega^i$.
3. Within an acceptable time frame, A sends $\omega'$ to B.
4. B records time-stamp $T'$, verifies that $\omega'$ is not already recorded and U has enough balance in the registered account.
5. B decrypts $C_2^{i+1}$ and gets
$$(i+1)||T^{i+1} = D_{ck}(E_{ck}((i+1)||T^{i+1}))$$
and checks time interval $T' - T^{i+1}$ is acceptable.
6. B verifies the signature by checking
$$(\sigma_{\alpha'})^e \stackrel{?}{\equiv} \mathcal{H}^{(i+1)}(\mathcal{H}^{r-(i+1)}(m)) \pmod{n}.$$
7. B deducts the amount $a^i$ from U's account $b_U$, adds it into M's account $b_M$, and informs A.

Including the $\mathcal{H}^{j-i}(m)$ value in each $i^{\text{th}}$ e-check tuple prevents e-check forgery. In order to resist e-check manipulation, payer should also compute $C_2^i$ as $C_2^i = E_{ck}(i||a^i||b^i||M||T^i)$ to bind the amount value $a$, random value $b$, and payee identity M which results in integrity assurance. However, even if these fixes are applied, the scheme would still fail to satisfy e-check anonymity requirement.

# 6 ANALYSIS OF CWL SCHEME

C.-L. Chen *et al.* claim that CCL scheme has some shortages such as time synchronizing issue and large computation overhead, (Chen et al., 2010). CWL scheme is proposed based on CCL and given in two phases, namely Registration and Paying.

## 6.1 CWL Protocol

In order to be consistent, here we again split the protocol into Initializing, Issuing, Paying and Depositing. This scheme requires setting a maximum value $W$ for the e-checkbook at the issuance phase which is used to make sure that the total face values of the e-checks belonging to the same e-checkbook does not exceed $W$.

Initializing

1. B chooses a one-way hash function $\mathcal{H}(\cdot)$.
2. B generates two large primes $p$ and $q$ and computes $n = pq$.
3. B selects a public key $e$ and computes $d$ such that $e \cdot d = 1 \pmod{\phi(n)}$,
4. B keeps $x_{\mathsf{B}} \leftarrow (p,q,d)$ as secret, and publishes public parameters $\mathsf{pp} \leftarrow (n,e,\mathcal{H}(\cdot))$.

Issuing

1. A payer registers with her identity U to a bank B, and shares a symmetric key $ck_{\mathsf{U}}$ for the symmetric encryption scheme $E$ over a secure channel.
2. B generates an identity $CID_{\mathsf{U}} = \mathcal{H}(\mathsf{U} \oplus d)$ with some random value $d$, computes $\alpha = \mathcal{H}(\mathsf{U} \oplus W \oplus r)$, and creates an e-checkbook $\omega_{\mathsf{U}} \leftarrow \{r, \alpha, \sigma_{CID_{\mathsf{U}}}, W\}$.
3. B stores $\{\mathsf{U}, CID_{\mathsf{U}}, ck_{\mathsf{U}}\}$ and sends $(CID_{\mathsf{U}}, r, \alpha, \sigma_{CID_{\mathsf{U}}}, W)$ to U over a secure channel.

Paying

1. U chooses $R, b, N_1, N_2$, computes $k \equiv R^{pk_{\mathsf{M}}}(b \oplus N_1) \pmod{n_{\mathsf{M}}}$ where $(n_{\mathsf{M}}, pk_{\mathsf{M}})$ being M's RSA public keys, and sends $k$ to M for signing blindly.
2. M computes $k' \equiv k^{sk_{\mathsf{M}}} \pmod{n_{\mathsf{M}}}$ and sends $k'$ to U.
3. U computes $M \equiv k'R^{-1} = (b \oplus N_1)^{sk_{\mathsf{M}}} \pmod{n_{\mathsf{M}}}$, $C_1 = \mathcal{H}(CID_{\mathsf{U}} \oplus a \oplus M)$ and $C_2 = E_{ck_{\mathsf{U}}}(r||a||N_2) \oplus \sigma_{CID_{\mathsf{U}}}$.
4. U sends $(C_1, C_2, a, b, N_1, N_2, CID_{\mathsf{U}}, \mathsf{B})$ to M.

Depositing

1. M verifies $C_1 \stackrel{?}{=} \mathcal{H}(CID_{\mathsf{U}} \oplus a \oplus ((b \oplus N_1)^{sk_{\mathsf{M}}} \pmod{n_{\mathsf{M}}}))$ and sends $(C_2, a, N_2, CID_{\mathsf{U}})$ to B over a secure channel.
2. B computes

$$(j, a, N_2) \leftarrow D_{ck_{\mathsf{U}}}(C_2 \oplus (CID_{\mathsf{U}}^{sk_{\mathsf{B}}} \pmod{n_{\mathsf{B}}}))),$$

and checks if $(j, a, N_2)$ is already stored in B's spent e-check database. If so B rejects e-check, otherwise stores $(j, a, N_2)$ as spent e-check.

3. B updates and assures remaining balance $W_{new} = W - a \geq 0$, remaining unspent e-checks number $r_{new} = r - 1 \geq 0$ and $\alpha_{new} = \mathcal{H}(CID_{\mathsf{U}} \oplus W_{new} \oplus r_{new})$ accordingly.
4. If all controls pass, B accepts the e-check, deducts the amount $a$ from U's account, adds it into M's account, computes $V_1 = (\mathsf{B} \oplus N_2)^{sk_{\mathsf{B}}} \pmod{n_{\mathsf{B}}}$, $V_2 = (W_{new} \oplus r_{new} \oplus N_3)^{pk_{\mathsf{M}}} \pmod{n_{\mathsf{M}}}$ for a random $N_3$, and returns $(V_1, V_2)$ to M and $(\alpha_{new}, W_{new}, r_{new}, N_3)$ to U over a secure channel.
5. M verifies $V_1^{pk_{\mathsf{B}}} \pmod{n_{\mathsf{B}}} \stackrel{?}{=} (\mathsf{B} \oplus N_2)$, computes $V_4 = \mathcal{H}(V_2^{sk_{\mathsf{M}}} \pmod{n_{\mathsf{M}}})$ and sends $V_4$ to U.
6. U verifies $V_4 \stackrel{?}{=} \mathcal{H}(W_{new} \oplus r_{new} \oplus N_3$ and $\alpha_{new} \stackrel{?}{=} \mathcal{H}(CID_{\mathsf{U}} \oplus W_{new} \oplus r_{new})$.

## 6.2 Flaws of CWL Scheme

The authors of CWL scheme claim that their scheme is secure against e-check forgery attack, see (Chen et al., 2010, Section 5.2: The forgery attack issue). The reasoning is based on the secrecy of $ck_{\mathsf{U}}$ which is the secret key of the symmetric-key encryption scheme $E$ shared between U and B.

As authors describe in (Chen et al., 2010); the registration phase, payee's depositing request to B, and B's final response are carried over a secure channel while rest of the communications are transmitted over an insecure channel. We now give an e-check manipulation attack which is similar to the given in Section 5.2.

1. An adversary A keeps track of U's transactions, records the tuple $(C_1, C_2, a, b, N_1, N_2, CID_{\mathsf{U}}, \mathsf{B})$ and blocks the e-check transmission that is meant to be a payment from the payer U to a payee M.
2. A sends $(CID_{\mathsf{U}}, N_2, a, C_2)$ to B as if it is a payment for herself over a secure channel.
3. As the $(CID_{\mathsf{U}}, N_2, a, C_2)$-tuple is created by U, B accepts the e-check, deducts the amount $a$ from U's account, adds it into the A's account, computes

$$V_1 = (\mathsf{B} \oplus N_2)^{sk_{\mathsf{B}}}, V_2 = (W_{new} \oplus r_{new} \oplus N_3)^{pk_{\mathsf{A}}},$$

and returns $(V_1, V_2)$ to A and $(\alpha_{new}, W_{new}, r_{new}, N_3)$ to U.
4. A computes $V_4 = \mathcal{H}(V_2^{sk_{\mathsf{A}}})$ and sends $V_4$ to U.
5. A can prove validity of the forged e-check by presenting the tuple $(C_1', C_2, a, b', N_1', N_2, CID_{\mathsf{U}}, \mathsf{B})$, where $C_1' = \mathcal{H}(CID_{\mathsf{U}} \oplus a \oplus (b' \oplus N_1')^{sk_{\mathsf{A}}})$ for some randomly chosen $b', N_1'$.

In order to resist this attack, a payer U should compute $C_2 = E_{ck_U}(r||a||b||ID_M||N_2||\mathcal{H}(C_1)) \oplus r_{CID_U}$, and send $(C_1, C_2, a, b, N_1, N_2, CID_U, B)$ to the payee M. M should also include $C_1$ to the tuple she sends to B after verifying

$$C_1 \stackrel{?}{=} \mathcal{H}(CID_U \oplus a \oplus ((b \oplus N_1)^{sk_M} \pmod{n_M})).$$

When B receives $(C_1', C_2, a, N_2, CID_U)$, computes

$$(j, a, b, ID_M, N_2, \mathcal{H}(C_1)) \leftarrow D_{ck_U}(C_2 \oplus (CID_U^{sk_B} \pmod{n_B}))$$

and checks if $\mathcal{H}(C_1) = \mathcal{H}(C_1')$. Rest of the depositing phase follows the same.

However, even if this fix is applied, the scheme does not fulfill the e-check anonymity requirement.

# 7 ANALYSIS OF CCW SCHEME

C.-L. Chang *et al.* proposed CCW e-checkbook scheme that utilizes elliptic curve cryptography, (Chang et al., 2016).

## 7.1 CCW Protocol

CCW scheme is proposed in two phases, namely Initialization and Paying. We split the protocol into Initializing, Issuing, Paying and Depositing. In this scheme, maximum amount $w$ that can be spent with an e-check needs to be determined at the issuance phase.

Initializing

1. Let $E_p$ be an elliptic curve points group over a finite field of characteristic $p$ and $\langle Q \rangle \in E_p$ be a subgroup of order $n$.
2. Let $h_1$ and $h_2$ be one-way hash functions.
3. Bank B, the payer C and the payee M creates the private keys by randomly selecting $b, c, m \in \mathbb{Z}_n^*$ and set public keys $B_{pub} = bQ$, $C_{pub} = cQ$ and $M_{pub} = mQ$, respectively.

Issuing

1. A payer C registers with her identity $ID_C$ to a bank B,
2. C chooses $k \in \mathbb{Z}_n^*$ and computes $K = kQ$, both are kept secret.
3. C sends $(ID_C, h_2(wK))$ to B for e-checkbook issuance.
4. B randomly chooses $t_B \in \mathbb{Z}_n^*$, computes $T_B = t_B Q$, sets $r = x_{T_B} \pmod{n}$ where $x_{T_B}$ is the x-coordinate of the point $T_B$ and creates a virtual identity $VID_c = h_1(t_B) \oplus ID_C$.
5. B then simultaneously computes

$$e = h_1(VID_c || h_2(wK)), s = t_B^{-1}(h_2(ejQ) + rb),$$
$$H_0 = h_1(t_B) \oplus h_2(bC_{pub}),$$

where $j$ is the number of e-checks in the e-checkbook.

6. B sends $(VID_C, (r,s), j, H_0)$ as e-checkbook where $(r,s)$ being its signature on the e-checkbook.
7. Upon receiving the e-checkbook $\omega_C$, C computes $VID_C = H_0 \oplus h_2(cB) \oplus ID_C$ and $e = h_1(VID_c || h_2(wK))$, verifies the signature and stores the e-checkbook as $(VID_C, (r,s), j, e)$.

Paying

1. C randomly selects $t_C \in \mathbb{Z}_n^*$ and hashes $h_1(t_C)$ to a point $X \in E_p$.
2. C computes $C_C = (t_C Q, X + t_C M_{pub}) = (X_1, Y_1)$ and sends $C_C$ to M.
3. M retrieves $X$ by computing $X = Y_1 - mX_1$, converts $X$ to $h_1(t_C)$ and hashes $h_1^2(t_C) = h_1(h_1(t_C))$ to a point $Y \in E_p$.
4. By selecting a random $t_M \in \mathbb{Z}_n^*$, M computes $C_M = (t_M Q, (Y + t_M h_1(t_c) Q)) = (X_2, Y_2)$ and send $C_M$ to the payee C.
5. Upon receiving $C_M$, similarly C retrieves $Y = Y_2 - h_1(t_C) X_2$ and checks the validity with $h_1(t_C)$, otherwise terminates.
6. If validity holds, C computes

$$F = (w-a)K + t_c X_2, \qquad H_1 = h_2(cB_{pub}) \oplus i,$$
$$H_2 = h_2(cB_{pub}) \oplus T_C \qquad H_3 = aK = akQ$$

where $a \leq w$ is the amount, $i$ is the number of e-checkbook has been used, $T_C$ is the timestamp.

7. C sends $(VID_C, \mathrm{ID}_B, a, j, (r,s), F, H_1, H_2, H_3, T_C)$ as the $i$-th e-check to the payee M.

Depositing

1. Upon receiving an e-check payment

$$(VID_C, \mathrm{ID}_B, a, j, (r,s), F, H_1, H_2, H_3, T_C),$$

M first retrieves $(w-a)K = F - t_M X_1$ and by using $(w-a)K$ and $H_3$, computes

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

2. Next, M verifies $r \stackrel{?}{=} x_2 \pmod{n}$.
3. If the equation holds, the M sends

$(VID_C, \mathrm{ID}_B, a, j, (r,s), F, H_1, H_2, H_3, T_C)$ to the B.

4. B retrieves C's identity $ID_C$ from $VID_C$, verifies C's account balance is enough for depositing.
5. B obtains $i$ and $T_C$ by computing
$$i = H_1 \oplus h_2(bC_{pub}) \text{ and } T_C = H_2 \oplus h_2(bC_{pub}).$$
6. Then, B 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.
7. Next, B computes
$$(h_2(e((j-i)Q+iQ))Q+rB_{pub})s^{-1} = (x_3, y_3)$$
to determine whether the equation $r \stackrel{?}{=} x_2 \pmod n$ holds or not.
8. If the equation holds, B deducts the amount $a$ from C's account and adds it into M's amount.
9. Finally B notifies both C and M.

### 7.2 Flaws of CCW Scheme

Here, we give an e-check manipulation attack which is similar to the given in Section 5.2. As already mentioned, at the payment phase, the payer first authenticates the payee and then sends the following tuple as the $i$-th e-check to the payee.

$$(\mathrm{VID}_C, \mathrm{ID}_B, a, j, (r,s), F, H_1, H_2, H_3, T_C),$$

where $\mathrm{VID}_c$ is the payer's virtual identity, $\mathrm{ID}_B$is ID of the bank, $a$ is the amount, ($a \le w$, $w$ max. amount), $j$ is maximum number of e-checks in the e-checkbook, $(r,s)$ is bank's signature on payer's e-checkbook,$T_C$ is timestamp,

$$F := (w-a)K + t_c X_2, \quad H_1 := h_2(cB_{pub}) \oplus i,$$
$$H_2 := h_2(cB_{pub}) \oplus T_C, \quad H_3 := aK = akQ.$$

As it can be noticed easily, changing the amount value $a$ requires changing $F$ and $H_3$ values. By computing

$$\begin{aligned} H_3' &= zH_3 = zaK, \text{ and} \\ F' &= F - (z-1)H_3 \\ &= (w-a)K + t_c X_2 - (z-1)aK \\ &= (w-za)K + t_c X_2, \end{aligned}$$

one can manipulate the e-check tuple as $(\mathrm{VID}_C, \mathrm{ID}_B, za, j, (r,s), F', H_1, H_2, H_3', T_C)$, for any $a < za \le w$. As a result, instead of $a$, the amount $za$ will be deducted from the payers account.

In fact, in a similar way, the e-check index $i$ value can also be altered by letting $H_1' = H_1 \oplus l$ such that $i < i + l \le j$.

In any e-checkbook scheme, integrity of the e-check index, amount and payee information should be satisfied. In this scheme, unfortunately, none of them is assured.

## 8 CONCLUSION

Although the volume of money transferred by checks is high, surprisingly there is not much mechanisms aim to transform the checkbook systems into fully digitalized e-checkbook systems. In this work, we analyzed security of the previously proposed e-checkbook schemes (Pasupathinathan et al., 2005; Chen et al., 2009; Chang et al., 2009; Chen et al., 2010; Chang et al., 2016) and showed that these propositions do not satisfy their claimed security by presenting practical attacks on each of them. These e-check forgery and/or e-check manipulation attacks can be evaded by assuring integrity and authenticity of the e-check index, the amount and the payee. Even if the corrections given in respective sections are applied, these schemes would still suffer efficiency and privacy-wise since they do not possess appropriate privacy preserving cryptographic building blocks for payment unlinkability and anonymity.

## REFERENCES


Anderson, M. M. (1998). The Electronic Check Architecture. Technical report, Financial Services Technology Consortium.

Brands, S. (1993). An Efficient Off-line Electronic Cash System Based On The Representation Problem. Technical report, Centrum Wiskunde & Informatica (CWI).

Chan, N. (2015). e-Cheque: A new era of payments in Hong Kong.

Chang, C.-C., Chang, S.-C., and Lee, J.-S. (2009). An online electronic check system with mutual authentication. *Computers & Electrical Engineering*, 35(5):757 – 763.

Chang, C.-C., Chang, S.-C., and Wu, Y.-C. (2016). Novel electronic check mechanism using elliptic curve cryptosystem. *Journal of Computers*, 27(3):111–122.

Chaum, D., den Boer, B., van Heyst, E., Mjølsnes, S., and Steenbeek, A. (1990a). Efficient offline electronic checks. In *EUROCRYPT '89*, pages 294–301. Springer.

Chaum, D., Fiat, A., and Naor, M. (1990b). Untraceable electronic cash. In *CRYPTO' 88*, pages 319–327. Springer.

Check 21 Act (2003). The Check Clearing for the 21st Century Act (Check 21).

Chen, C.-L., Wu, C.-H., and Lin, W.-C. (2010). Improving an on-line electronic check system with mutual authentication. In *AIT 2010*.

Chen, T.-H., Yeh, S.-C., Liao, K.-C., and Lee, W.-B. (2009). A practical and efficient electronic checkbook. *Journal of Organizational Computing and Electronic Commerce*, 19(4):285–293.

Chen, W.-K. (2005). Efficient on-line electronic checks. *Applied Mathematics and Computation*, 162(3):1259 – 1263.

Katz, J. and Lindell, Y. (2014). *Introduction to Modern Cryptography*. Chapman & Hall/CRC, 2nd edition.

Pasupathinathan, V., Pieprzyk, J., and Wang, H. (2005). Privacy enhanced electronic cheque system. In *7th IEEE International Conference on E-Commerce Technology (CEC'05)*, pages 431–434.

Sertkaya, I. and Kalkar, O. (2019). An efficient electronic checkbook scheme with mutual authentication. *Suleyman Demirel University Journal of Natural and Applied Sciences*, pages 590 – 596.

Sertkaya, I. and Kalkar, O. (2021). A privacy enhanced transferable electronic checkbook scheme. *Wireless Personal Communications*, pages 1–27.