IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 23, 2018, accepted September 30, 2018, date of publication October 5, 2018, date of current version October 31, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2874021

A Robust Mobile Payment Scheme With Smart Contract-Based Transaction Repository

KUO-HUI YEH[1,2], (Senior Member, IEEE), CHUNHUA SU[2], JIA-LI HOU[1], WAYNE CHIU[1], AND CHIEN-MING CHEN[3]

[1]Department of Information Management, National Dong Hwa University, Hualien 97401, Taiwan
[2]Division of Computer Science, The University of Aizu, AizuWakamatsu 965-8580, Japan
[3]College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

Corresponding author: Chien-Ming Chen (chienming.taiwan@gmail.com)

This work was supported in part by JSPS KAKENHI under Grant Kiban C 18K11298, in part by the Ministry of Science and Technology, Taiwan, under Grant MOST 105-2221-E-259-014-MY3, Grant MOST 105-2221-E-011-070- MY3, Grant MOST 105-2923-E-182-001-MY3, and Grant MOST 107-2218-E-011-012.

ABSTRACT Recently, the popularity and universality of smart-devices has led to rapid advancement in the development of applications for mobile commerce around the world. Novel mobile payment schemes, such as Apple pay, Android pay, and Samsung pay are becoming an increasingly popular ways to conduct online transactions, no matter what type of smart devices are used. Due to the attendant growth in the importance of security, significant attention has been devoted to the challenge of designing and implementing a robust mobile payment scheme for securing online transactions. In this paper, we demonstrate a robust mobile payment scheme based on sturdy certificateless signatures with bilinear pairing. We elegantly refine the proposed mobile payment scheme to make it suitable for computation-constrained mobile devices. The practicability of the proposed mobile payment scheme is then certified via a rigorous security analysis and thorough performance evaluation using the Raspberry PI as the implementation platform for our proposed scheme. Furthermore, we implement a transaction repository with the aid of smart contract technology. The simulation results, based on Ethereum, demonstrate the feasibility of employing the smart contract technology to secure mobile payments.

INDEX TERMS Bilinear pairing, certificateless signature, mobile payment, security, smart contract.

I. INTRODUCTION

Due to the rapid advancement of communication technologies and the universality of smart-devices, myriad applications have been developed for smart-devices (including smartphones and smart objects) to allow developers to provide various types of commercial and marketing services and open new revenue streams in the process. Customers have gradually shown a shift in the way they make purchases, moving away from the traditional use of credit cards and opting instead toward new methods, such as mobile payments processed on intelligent handheld devices. The obvious advantage is that an online payment transaction can be initiated and completed anytime, anywhere, via smart-device. The first micro-payment scheme was proposed by Rivest [10] in 1996. Nowadays, with the growing interest in crypto-currencies and decentralized payment systems (e.g., Bitcoin proposed by Nakamoto, 2008 [7]), mobile payment technology is anticipated to become one of the most important form of killer application in the sphere of mobile commerce. Recently, reaping the benefits of the mobile payment market has been a major focused of industry, and several mobile payment schemes, such as ApplePay, Android Pay, Samsung Pay and Line Pay, have swept into fashion. These techniques have fundamentally changed the way individuals' think about their payment behavior, from ‘‘static, immovable and physical'' to ‘‘dynamic, electronic, moveable and virtual''. Customers have become enamored with the undeniable convenience of making online payments. However, such ‘‘convenience'' is always accompanied by security threats and privacy-disclosure risks, especially with respect to the security of hardware, software, and communication architecture. Potential threats to existing mobile payment techniques include (a) fraud accomplished through phishing or social engineering, (b) misuse of stolen mobile devices, (c) inherent vulnerabilities of payment applications and operating system access permissions, (d) various attacks, i.e. relay attack, man-in-the-middle (MITM) attack, and denial of services (DoS) attack, used to interfere with service availability, and (e) interested parties' and cardholder's sensitive data being compromised [20]. Hence, security is of utmost importance and is absolutely indispensable for mobile payments. It remains, unfortunately, in the early stages of development, owing to the insufficiency of the support provided by existing hardware and software techniques. Furthermore, room for improvement exists for current mobile payment techniques, particularly from the standpoint of standards and interoperability. More specifically, widely-accepted standards for mobile payment are still lacking, and this impedes the development of interoperability among systems. Based on the above observations, we are motivated to propose a secure and privacy-aware mobile payment scheme which is able to protect the user's sensitive data, i.e. personal private information, account number, and relevant credit card numbers, from being disclosed and/or tampered with when such data is processed and transferred via mobile payment applications. In addition, the proposed scheme must fulfill the following security requirements:

- Data confidentiality is critical for payment transactions among all involved entities, such as mobile clients, trusted third parties, payment service providers (or platforms) and merchants. Payment data is sentitive and closely related to personal preferences, making privacy important. Accordingly, preventing payment data from being disclosed (or compromised) by malicious adversaries during transmission is paramount for mobile payment.
- Transaction non-repudiation is also critical for mobile payment. When disputes over transactions (or their details) arise between mobile clients and merchants, the non-repudiation of said transactions becomes indispensable for the purpose of adjudicating disputes, and it is also valuable for auditing purposes.
- During communication among payment-involved entities, a malicious adversary can trick the entities with tampered and counterfeited messages (and connections) in various ways. First, man-in-the-middle (MITM) attacks and impersonation attacks launched by malicious adversaries are usually accompanied with fake messages intended to trick victims (who are legitimate entities) into believing the adversaries are legal payment entities and therefore not questioning the adversaries' "legal but malicious" behaviors. Second, a legally transmitted message may be eavesdropped (or even interrupted) by adversaries, and be reused from session to session to cheat the legitimate entity victims. Since the message in question is legally issued by legitimate entities, its legitimacy will be verified if no further checking mechanism for replay behaviors exists. Third, Denial of Service (DoS) attacks may be detrimental to the transaction process of a payment protocol. Adversaries usually send legal messages to communicating entities and exhaust the victims' computation and power resources. The result is that the service availability of the payment protocol cannot be guaranteed. Based on the foregoing observations, it is important to integrate a verification mechanism to confirm the legitimacy of communicating entities in a secure payment transaction scheme. That is to say, a robust payment scheme must guarantee to be immune against MITM, impersonation, replay and DoS attacks.

In this study, we propose a robust mobile payment mechanism to guarantee security during online transactions. The proposed mobile payment mechanism consists of sturdy crypto-primitives, i.e. certificateless and bilinear pairing. In Section 2, we introduce the state-of-the-art of mobile payment research and discuss the most relevant studies. Then, we introduce the processes of our proposed mobile payment mechanism in detail in Section 3. Next, Section 4 sets out the security analysis and performance evaluation of the proposed scheme as a practicability examination. Finally, we present the concluding remarks in Section 5.

II. RELATED WORK

In 2015, Daza *et al.* [3] introduced an off-line micro-payment scheme, called FRoDO, to guarantee the security when customers and vendors are disconnected from the network. They proposed FRoDO as a resilient point-of-sale system in which two core elements, i.e. a coin element and an identity element, are established to support a two-factor authentication to the customer by strengthening the relationship between the devices. Finally, the authors analysed the proposed micropayment scheme to support their claims of effectiveness and viability. Next, Garg and Garg [4], proposed a payment scheme with biometric and tokenization techniques. The presented method allows users to make payments at merchant terminals by inputting their fingerprints. This design eliminates the need to carry multiple physical cards when making payments. However, the implementation of their proposed biometric-based payment mechanism presents a hurdle in terms of the secure storage of human biometrics. Later, Jetsiktat *et al.* [6], demonstrated the possibility of integrating biometric verification into the online payment process, whereby the user's face is efficiently captured and analysed for user verification. A visual standard called MPEG7-EHD and developed by Moving Picture Experts Group is exploited to support the face matching procedures. An average value of i.e. 99% verification accuracy was obtained in the simulation conducted by the authors.

After that, Yang [15], first pointed out four threats to mobile payment. They are: (a) lack of encryption at mobile terminals, (b) lack of user authentication, (c) possible security threats, i.e. transaction repudiation, malicious overdraft and deceptive business, which may originate from a vulnerable credit management system, and (d) lack of protective measures for securing payment applications on mobile terminals. To address and overcome these threats, the authors introduced a mobile payment mechanism with identity-based cryptography. Furthermore, to investigate the practicability of the presented idea, the authors deployed the payment system on a 32-bit high-performance security chip, i.e. WIS08SD548E with a secure digital memory card. They claimed that the proposed method possesses simplicity, reliability and efficiency when compared to other existing methods, such as SET and 3-D Secure. In 2016, Park and Lee [8] identified a privacy infringement risk during the operation of NFC technology. They presented an attack scenario to demonstrate how credit card information could be hacked using a POS machine with KS X 6928 standard. In order to eliminate this risk, the authors integrated the technique of signature record type definition (RTD) which constitutes part of the NFC standard into the mobile payment process. The presented method provides scalability for what are currently NFC-based mobile payments. Then, Urien [12] introduced a mobile payment infrastructure based on open standards and protocols. The proposed infrastructure relies on Host Card Emulation (HCE), a secure elements technique (e.g., EMV cards), a remote access technique (e.g., RACS protocol) and a secure protocol (e.g., TLS). Nevertheless, the author reported the issue of system scalability as a direction for future work.

Later, to investigate the practicability of integrating the mobile payment concept into the cloud, Urien and Aghina [13] presented a mobile payment system with open technologies. The system combines the techniques of the Android platform, NFC and TLS. At the same year, Urien and Aghina [14] further demonstrated a similar implementation, i.e. the so-called SIMulation project, which involved construction of a mobile payment platform with cloud services, in which smartphones act as a logical bridge during communications between merchants and the payment platform itself. In addition, the communication security is guaranteed via a SIM module plugged into the TLS protocol. Meanwhile, focusing on the hardware security side of things, Chen and Zheng [2] proposed a circuit design for a low power range-controlled communication chip for mobile payment. In its implementation, signals are transmitted via low frequency electromagnetic field, and a three-stage gain amplifier is exploited to amplify the signals. Moreover, the authors presented their designs for digital-to-analog converters and comparators to support the verification of payment transactions. Furthermore, Zheng *et al.* [16] introduced a mobile payment framework called TrustPAY, designed to guarantee transaction authenticity, privacy protection, data confidentiality, code integrity and service availability. In addition, they implemented a prototype system on an ARM CoreTile Express A9x4 using an ARM Fast-Model with open virtualization software stack for the ARM TrustZone. The authors then presented their analyses based on various security and privacy scenarios to evaluate the practicability and scalability of the presented idea. In another work, with the support of the secure element embedded inside a NFC phone, Turk and Cosar [11] proposed a payment method to provide proprietary payments and identification. The authors then explained the integration of their proposed model and possible proprietary payment scenarios, such as public transport payment, to demonstrate the its practical use. They further provided an open protocol as a secure NFC payment and identification scheme. However, the presented method remains at the conceptual level, without any implementation (or evaluation).

III. THE PROPOSED MOBILE PAYMENT SCHEME

In this section, we introduce our proposed mobile payment scheme for online transactions. Before that, the system parameters, including the elliptic curve and bilinear pairing, are presented.

First, we set the notation $^{E}\!/_{E_p}$ as an elliptic curve E over a prime finite field E_p. It is defined by the equation $y^2 = x^3 + ax + b$, where $a, b \in F_p$ are constants such that $\Delta = 4a^3 + 27b^2 \neq 0$ $\Delta = 4\mathrm{a}^3 + 27\mathrm{b}^2 \neq 0$. All points $P_i = (x_i, y_i)$ on E and the infinity point O form a cyclic group G and are under the operation of point addition $R = P + Q$ with the chord-and-tangent rule. In that case, $t \cdot P = P + P + \ldots + P$ (t times) is considered as scalar multiplication, where P is a generator of G with a prime order n. After that, we define the Elliptic Curve Discrete Logarithm Problem (ECDLP) as follows: *given a group G of elliptic curve points with a prime order n*, a generator P of G and a point $x \cdot P$, it is computationally infeasible to derive x, where $x \in Z_n^*$. Next, set G_1 and G_2 as the cyclic group with the a prime order q where G_1 is an additive cyclic group and G_2 is a multiplicative cyclic group. Let a mapping of $e : G_1 \times G_1 \to G_2$ hold the following conditions. They are (a) bilinear: $\forall a, b \in Z_q^*, \forall P, Q \in G_1 : e\,(aP, bQ) = e\,(P, Q)^{ab}$; (b) non-degenerate: $e \neq 1$; and (c) computability: this means that an efficient algorithm exists to compute *ae*.

The security of our proposed mobile payment scheme relies on the difficulty of breaking a bilinear pairing and the intractability of the ECDLP. During the system initialization phase, the following steps are launched. Given a secure parameter k, a Trusted Third Party (TTP) selects two groups G_1 and G_2 with the same prime order q and a bilinear pairing e:$G_1 \times G_1 \to G_2$, where P is a generator of G_1. Then, TTP generates a random number $s \in Z_q^*$ as its private key and computes its public key as $PK_{TTP} = s \cdot P$. Next, TTP determines three robust one-way hash functions, i.e. H_1:$\{0, 1\}^* \times G_1 \to Z_q^*$, H_2:$\{0, 1\}^* \times G_1 \times G_1 \to Z_q^*$ and H_3:$\{0, 1\}^* \times G_1 \times G_1 \times G_1 \to Z_q^*$. After that, TTP publishes a set of public parameters, i.e. $params = G_1, G_2, q, e, P, PK_{TTP}, H_1, H_2, H_3, e(P, P)$. Meanwhile, the user U_i chooses a secret key x_i and computes the corresponding public key is $PK_i = x_i \cdot P$, and the merchant calculates its public key $PK_M = x_M \cdot P$ with a chosen secret key that is x_M.

A. THE PROCEDURES OF A NORMAL MOBILE PAYMENT OPERATION

Step 1: When a transaction starts, the user U_i sends a non-sensitive datum $NSD = (TID, AM)$ to TTP, where TID is the identity of the current transaction invoked by U_i and the amount, i.e. AM, of this transaction.

Step 2: Once TTP receives NSD, it selects a random number r_i and computes $R_i = r_i \cdot P$, $h_i = H_1(ID_i, R_i, PK_{TTP}, NSD)$,

$s_i = r_i + h_i \cdot s \bmod q$ and $\sigma_{i\_1} = s_i^{-1} \cdot P$. Next, *TTP* sends a response (s_i, R_i, σ_{i_1}) back to U_i. After receiving (s_i, R_i, σ_{i_1}), U_i checks its validity with the following computations: (a) compute $h_i = H_1(ID_i, R_i, PK_{TTP}, NSD)$ and (b) check if $e\left(\sigma_{i\_1}, R_i + h_i \cdot PK_{TTP}\right) = e(P, P)$ holds. The correctness of $e\left(\sigma_{i\_1}, R_i + h_i \cdot PK_{TTP}\right) = e(P, P)$ is presented below.

$$\begin{aligned} &e\left(\sigma_{i\_1}, R_i + h_i \cdot PK_{TTP}\right)\\ &\quad = e\left(s_i^{-1} \cdot P, r_i \cdot P + h_i \cdot s \cdot P\right)\\ &\quad = e\left(s_i^{-1} \cdot P, (r_i + h_i \cdot s) \cdot P\right)\\ &\quad = e\left(s_i^{-1} \cdot P, s_i \cdot P\right)\\ &\quad = e\left(P, P\right)^{s_i^{-1} s_i}\\ &\quad = e(P, P) \end{aligned}$$

If the examination holds, U_i believes the validity of (s_i, R_i, σ_{i_1}). The above procedures refer to Figure 1. After the payment is authorized, U_i chooses a random number $r_1 \in Z_q^*$, and computes $H_3(r_1 \cdot x_i \cdot PK_M)$ and $EPD = H_3\left(r_1 \cdot x_i \cdot PK_M\right) \oplus PD$, where *SD* is the sensitive data, e.g., credit card number and personal private data relevant to U_i, and $PD = (NSD, SD)$ is the payment data of this transaction operation. Then, U_i computes $k_i = H_2(ID_i, PK_i, R_i, PK_{TTP}, EPD)$ and $\sigma_{i\_2} = (k_i \cdot s_i + x_i)^{-1} \cdot P$. These processes can be referred to Figure 2.

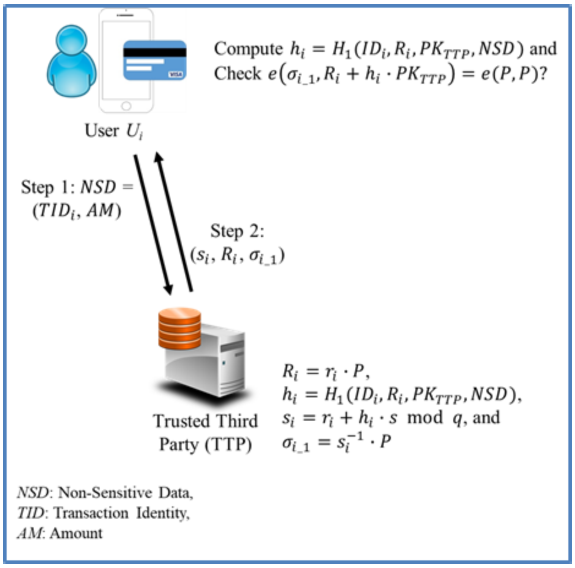

Compute $h_i = H_1(ID_i, R_i, PK_{TTP}, NSD)$ and
Check $e\left(\sigma_{i\_1}, R_i + h_i \cdot PK_{TTP}\right) = e(P,P)$?
User U_i
Step 1: $NSD = (TID_i, AM)$
Step 2: (s_i, R_i, σ_{i_1})
Trusted Third Party (TTP)
$R_i = r_i \cdot P$,
$h_i = H_1(ID_i, R_i, PK_{TTP}, NSD)$,
$s_i = r_i + h_i \cdot s \mod q$, and
$\sigma_{i\_1} = s_i^{-1} \cdot P$
NSD: Non-Sensitive Data,
TID: Transaction Identity,
AM: Amount

FIGURE 1. **Steps 1 and 2 of the proposed transaction mechanism.**

Step 3: U_i issues $(ID_i, TID, r_1, EPD, R_i, \sigma_{i\_2})$ as a transaction request to the mobile payment platform, as shown in Figure 3. Upon receiving $(r_1, EPD, R_i, \sigma_{i\_2})$, the mobile payment platform performs a re-encryption operation on *EPD*. That is, the payment platform randomly generates a number t_i, and computes $T_i = t_i \cdot P$, $Q_i = H_3(ID_{MPP}, TID)$, $Hash_i = H_3(t_i, EPD)$ and $Cipher_1 = t_i \oplus H_3\left(e\left(Q_i, t_i \cdot PK_M\right)\right)$, where ID_{MPP} is the identity of the mobile payment platform.

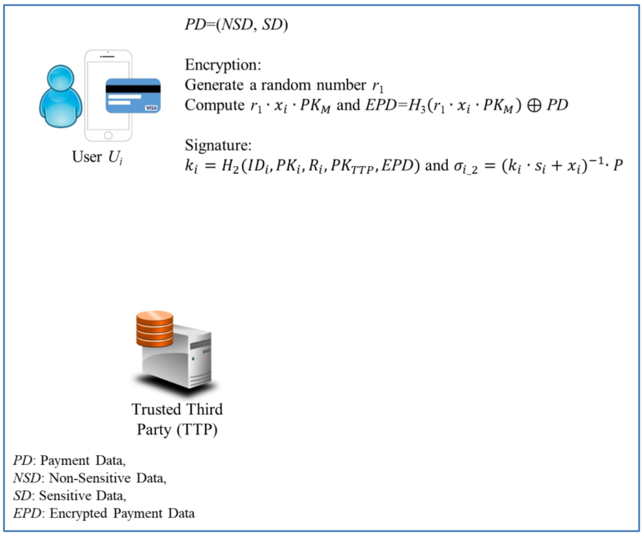

PD=(NSD, SD)
Encryption:
Generate a random number r_1
Compute $r_1 \cdot x_i \cdot PK_M$ and EPD=$H_3(r_1 \cdot x_i \cdot PK_M) \oplus PD$
Signature:
$k_i = H_2(ID_i, PK_i, R_i, PK_{TTP}, EPD)$ and $\sigma_{i\_2} = (k_i \cdot s_i + x_i)^{-1} \cdot P$
User U_i
Trusted Third Party (TTP)
PD: Payment Data,
NSD: Non-Sensitive Data,
SD: Sensitive Data,
EPD: Encrypted Payment Data

FIGURE 2. **The encryption and signature generation processes at the user side.**

Step 4: The mobile payment platform sends $(ID_i, TID, r_1, EPD, R_i, \sigma_{i\_2})$ and $(T_i, Q_i, Hash_i, Cipher_1)$ with ID_{MPP} to the merchant. First, the merchant calculates $x_M \cdot T_i$, $Q_i = H_3(ID_{MPP}, TID)$ and $H_3\left(e\left(Q_i, x_M \cdot T_i\right)\right)$, and retrieves t_i via the computation of $Cipher_1 \oplus H_3\left(e\left(Q_i, x_M \cdot T_i\right)\right)$. Then, the merchant checks the validity of t_i and *EPD* via the examination of the correctness of $H3(t_i, EPD)$=$Hash_i$.

Furthermore, the merchant computes $h_i = H_1(ID_i, R_i, PK_{TTP})$ and $k_i = H_2\left(ID_i, PK_i, R_i, PK_{TTP}, EPD\right)$, and checks the correctness of $e(\sigma_{i\_2}, k_i \cdot (R_i + h_i \cdot PK_{TTP}) + PK_i) = e(P, P)$. If the correctness examination holds, the merchant believes the validity of σ_{i_2}.

$$\begin{aligned} &e\left(\sigma_{i\_2}, k_i \cdot (R_i + h_i \cdot PK_{TTP}) + PK_i\right)\\ &\quad = e\left((k_i \cdot s_i + x_i)^{-1} \cdot P, k_i \cdot (r_i \cdot P + h_i \cdot s \cdot P) + x_i \cdot P\right)\\ &\quad = e\left((k_i \cdot s_i + x_i)^{-1} \cdot P, (k_i \cdot (r_i + h_i \cdot s) + x_i) \cdot P\right)\\ &\quad = e\left((k_i \cdot s_i + x_i)^{-1} \cdot P, (k_i \cdot s_i + x_i) \cdot P\right)\\ &\quad = e\left(P, P\right)^{(k_i \cdot s_i + x_i)^{-1}(k_i \cdot s_i + x_i)}\\ &\quad = e(P, P) \end{aligned}$$

After checking the validity of σ_{i_2}, the merchant retrieves *PD* via the computation of $H3(r_1 \cdot x_M \cdot PK_i)$. Note that *PD* contains *SD*, such as a credit card number and personal private data, relevant to U_i. With *PD*, the merchant is able to process the payment launched by U_i. The above procedures refer to Figure 4. Finally, as shown in Figure 5, the merchant will send a result, i.e. success or failure, of the current transaction operation back to U_i.

B. THE PROCEDURES OF A TRANSACTION LOG BEING UPLOADED TO A SMART CONTRACT-BASED REPOSITORY (I.E. FIGURE 6)

Upon sending a successful result to U_i, the merchant collects the critical information, i.e. $Tran_i = (TID_i, r_1, EPD, R_i, \sigma_{i\_1}, \sigma_{i\_2}, result)$, of the current transaction which

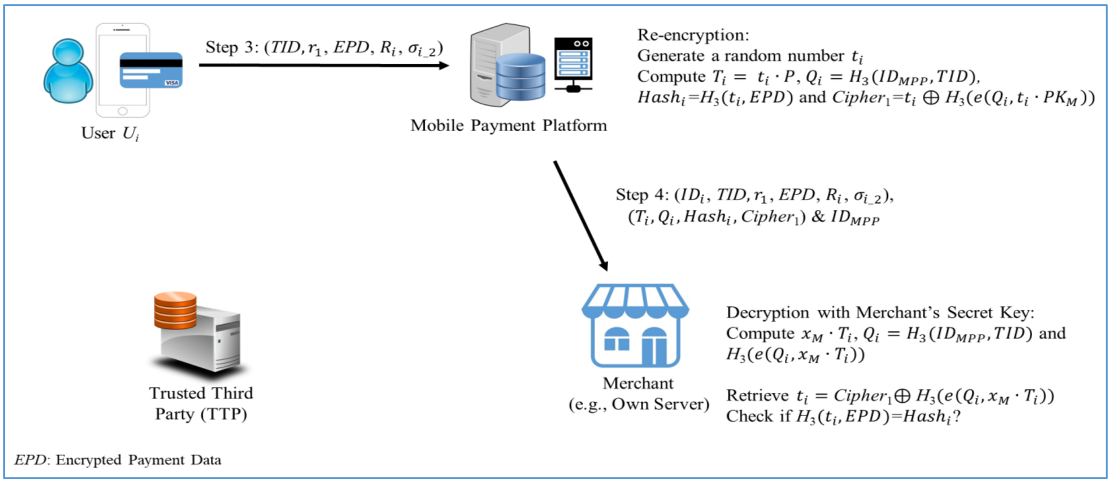

Step 3: $(TID, r_1, EPD, R_i, \sigma_{i\_2})$
User U_i
Mobile Payment Platform
Re-encryption:
Generate a random number t_i
Compute $T_i = t_i \cdot P$, $Q_i = H_3(ID_{MPP}, TID)$,
$Hash_i{=}H_3(t_i, EPD)$ and $Cipher_1{=}t_i \oplus H_3(e(Q_i, t_i \cdot PK_M))$
Step 4: $(ID_i, TID, r_1, EPD, R_i, \sigma_{i\_2})$,
$(T_i, Q_i, Hash_i, Cipher_1)$ & ID_{MPP}
Trusted Third
Party (TTP)
Merchant
(e.g., Own Server)
Decryption with Merchant's Secret Key:
Compute $x_M \cdot T_i$, $Q_i = H_3(ID_{MPP}, TID)$ and
$H_3(e(Q_i, x_M \cdot T_i))$
Retrieve $t_i = Cipher_1 \oplus H_3(e(Q_i, x_M \cdot T_i))$
Check if $H_3(t_i, EPD){=}Hash_i$?
EPD: Encrypted Payment Data

FIGURE 3. Steps 3 and 4 of the proposed transaction mechanism.

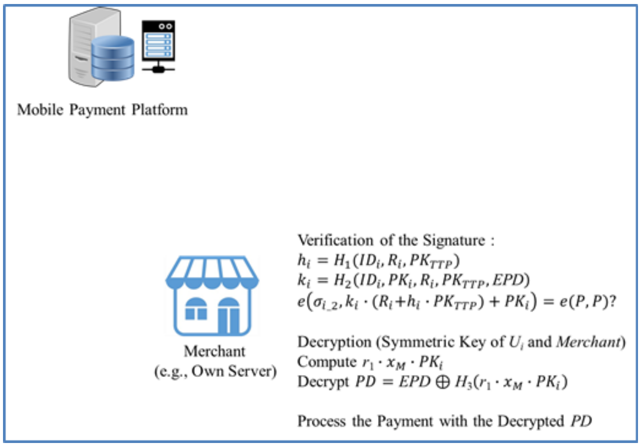

Mobile Payment Platform
Merchant
(e.g., Own Server)
Verification of the Signature :
$h_i = H_1(ID_i, R_i, PK_{TTP})$
$k_i = H_2(ID_i, PK_i, R_i, PK_{TTP}, EPD)$
$e\big(\sigma_{i\_2}, k_i \cdot (R_i{+}h_i \cdot PK_{TTP}) + PK_i\big) = e(P, P)$?
Decryption (Symmetric Key of U_i and *Merchant*)
Compute $r_1 \cdot x_M \cdot PK_i$
Decrypt $PD = EPD \oplus H_3(r_1 \cdot x_M \cdot PK_i)$
Process the Payment with the Decrypted *PD*

FIGURE 4. The signature verification and decryption processes at the merchant side.

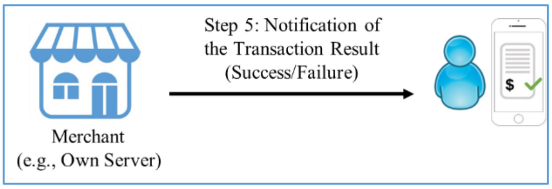


FIGURE 5. Notification of the transaction result.

has been successfully completed. Then, the merchant sends $Tran_i$ to TTP, and asks for a valid signature for this transaction. TTP then looks for r_i corresponding to TID_i, and computes $l_i = H_2(Tran_i, PK_{TTP})$, $s_i^{'} = r_i{+}l_i{\cdot}s \bmod q$, and $\sigma_{i\_3} = (s_i^{'})^{-1}{\cdot}P$. After that, TTP sends (r_i, σ_{i_3}) back to the merchant, which will soon forward $(r_i, \sigma_{i\_3}, Tran_i)$ to our proposed private Blockchain network. The Blockchain network then stores the incoming transaction log as a smart contract for the purpose of auditing. In that case, when transaction disputes happen between the user and the merchant, each interested party is able to retrieve $(r_i, \sigma_{i\_3}, Tran_i)$ corresponding to the target transaction, and verify it based on whether the correctness of $e\left(\sigma_{i\_3}, R_i + l_i \cdot PK_{TTP}\right) = e(P, P)$ holds or not. Note that $l_i = H_2(Tran_i, PK_{TTP})$ is computed before the verification.

$$\begin{aligned} & e\left(\sigma_{i\_3}, R_i + l_i \cdot PK_{TTP}\right) \\ & \quad = e\left(\left(s_i^{'}\right)^{-1} \cdot P., r_i \cdot P + l_i \cdot s \cdot P\right) \\ & \quad = e\left(\left(s_i^{'}\right)^{-1} \cdot P, \left(r_i + l_i \cdot s\right) \cdot P\right) \\ & \quad = e\left(\left(s_i^{'}\right)^{-1} \cdot P, \left(s_i^{'}\right) \cdot P\right) \\ & \quad = e\left(P, P\right)^{\left(s_i^{'}\right)^{-1} \times \left(s_i^{'}\right)} \\ & \quad = e(P, P) \end{aligned}$$

IV. SECURITY AND PERFORMANCE ANALYSES

This section demonstrates the security analysis of our proposed mobile payment scheme. In terms of the employment of certificateless signature and bilinear pairing cryptoprimitives, the analysis is accordingly based on the adversary and security models defined in the studies proposed by Huang *et al.* [5] and Al-Riyami and Paterson [1].

A. ADVERSARIES AND ORACLES

In a normal transaction operation, there exist two kinds of adversaries, i.e. Type I adversary A_I and Type II adversary A_{II}. Basically, adversary A_I (anyone except the KGC) possesses the ability to replace the user's public keys, such as PK_i and PK_M. However, A_I is not given with (s_i, R_i). On the other hand, adversary A_{II} has the master private key, i.e. s, of TTP, but A_{II} cannot replace any user's public key. In general, A_I and A_{II} can access the following three oracles:

➢ CreateUser: Given a query $ID \in \{0, 1\}^*$, the oracle gets (s_{ID}, R_{ID}), x_{ID} and PK_{ID}. Then, it adds a record of $(ID, (s_{ID}, R_{ID}), x_{ID}, PK_{ID})$ to the list L. Finally, PK_{ID} is returned.

➢ PublicKeyReplace: Given a query $(ID, PK_{ID}^{'})$, the oracle is able to replace the user's public key PK_{ID} with a new one $PK_{ID}^{'}$, and to update the list L.

➢ SecretValueExtract: Given a query $ID \in \{0, 1\}^*$, the oracle looks for the secret value x_{ID} in the list L, and

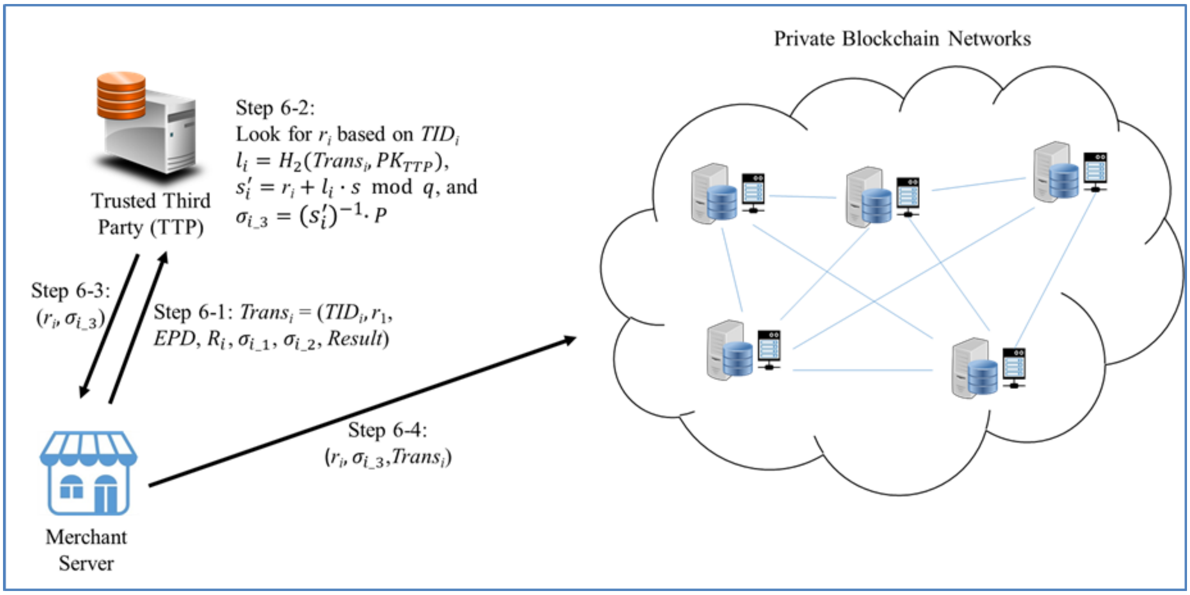

Private Blockchain Networks
Step 6-2:
Look for r_i based on TID_i
$l_i = H_2(Trans_i, PK_{TTP})$,
$s_i' = r_i + l_i \cdot s \mod q$, and
$\sigma_{i\_3} = (s_i')^{-1} \cdot P$
Trusted Third Party (TTP)
Step 6-3: (r_i, σ_{i_3})
Step 6-1: $Trans_i = (TID_i, r_1, EPD, R_i, \sigma_{i\_1}, \sigma_{i\_2}, Result)$
Step 6-4: $(r_i, \sigma_{i\_3}, Trans_i)$
Merchant Server

FIGURE 6. Transactions being uploaded to a Blockchain-based repository.

returns x_{ID}. Note that the secret value x'_{ID} corresponding to the replaced public key PK'_{ID} cannot be extracted.

Furthermore, Type I and II adversaries can also be divided into three classifications, i.e. normal adversary, strong adversary and super adversary, based on their power level. A normal adversary usually is able to learn a valid verification message, while a strong adversary has the ability to replace any user's public key and forge a valid verification message. The most powerful adversary, i.e. a super-level adversary, can learn valid verification messages for a replaced public key without any submission. It is believed that the highest security density is able to be guaranteed if the proposed payment scheme is robust against the super-level adversary. Hence, in the following we will investigate the security against the super type I and II adversaries in our proposed scheme.

- Game 1: Security against a super type I adversary

As the certificateless signature is adopted as the core security technique in our proposed transaction scheme, the robustness of our proposed scheme is thus majorly based on the existential unforgeability of the signatures generated in the transaction scheme. Therefore, we make the following statement. A super type I adversary A_I can obtain a signature σ_i such that $true \leftarrow Verify(m, \sigma_i, params, ID, PK_{ID})$ under the public key PK_{ID} chosen by A_I itself. The existential unforgeability of the certificateless signature in our proposed transaction scheme against a super type I adversary A_I is defined by the following games:

Phase 1: The challenger invokes the system initialization and returns the system parameters *params* to A_I.

Phase 2: A_I can adaptively access the oracles, i.e. CreateUser, PublicKeyReplace and SecretValueExtract, and can also access the PrivateKeyExtract oracle and SuperSign oracle.

➢ PrivateKeyExtract: Given a query *ID*, the oracle browses the list L and returns (s_{ID}, R_{ID}).

➢ SuperSign: Given a query (ID, m), the oracle returns a signature σ_i satisfying $true \leftarrow Verify(m, \sigma_i, params, ID, PK_{ID})$, where m denotes the message to be signed.

Phase 3: After all necessary queries, A_I outputs a forgery (m^*, σ_i^*, ID^*). After that, A_I wins the game if the outputted forgery satisfies the following conditions:

➢ A_I has never submitted (ID^*, m^*) to the SuperSign oracle;

➢ A_I has never submitted ID^* to the PrivateKeyExtract oracle;

➢ $true \leftarrow Verify(m, \sigma_i, params, ID, PK_{ID^*})$

We define the success probability of a super type I adversary A_I winning the above game as $Succ_{A_I}$, and the corresponding definition is made.

Definition 1: Our proposed transaction mechanism is secure against a $(t, q_{CU}, q_{PKR}, q_{SVE}, q_{PKE}, q_{SS})$ super type I adversary A_I if A_I runs in polynomial time t, makes at most q_{CU} times the CreateUser oracle query, q_{PKR} times the PublicKeyReplace oracle query, q_{SVE} times the SecretValueExtract oracle query, q_{PKE} times the PrivateKeyExtract oracle query, q_{SS} times the SuperSign oracle query, and $Succ_{A_I}$ is negligible.

- Game 2: Security against a super type II adversary

The type II adversary A_{II} simulates the *TTP* who possesses the master secret key s, and possibly engages in attack activities, including passive eavesdropping or launching signing queries. The existential unforgeability of the certificateless signature in our proposed transaction scheme against a super type II adversary A_{II} is defined by the following games:

Phase 1: The challenger invokes the system initialization and returns the system parameters *params* to A_{II}.

Phase 2: A_{II} can adaptively access the oracles, i.e. CreateUser, PublicKeyReplace, SecretValueExtract and SuperSign.

Phase 3: After all necessary queries, A_{II} outputs a forgery (m^*, σ_i^*, ID^*). After that, A_{II} wins the game if the outputted forgery satisfies the following conditions:

- ➢ A_{II} has never submitted (ID^*, m^*) to the SuperSign oracle;
- ➢ A_{II} has never submitted ID^* to the SecretValueExtract oracle;
- ➢ $true \leftarrow Verify(m, \sigma_i, params, ID^*, PK_{ID^*})$, where PK_{ID^*} is the original public key returned by the oracle CreateUser.

The success probability of a super type II adversary A_{II} winning the above game is defined as $Succ_{A_{II}}$, and the corresponding definition is made.

Definition 2: Our proposed transaction mechanism is secure against a $(t, q_{CU}, q_{PKR}, q_{SVE}, q_{SS})$ super type II adversary A_{II} if A_{II} runs in polynomial time t, makes at most q_{CU} times the CreateUser oracle query, q_{PKR} times the PublicKeyReplace oracle query, q_{SVE} times the SecretValueExtract oracle query, q_{SS} times the SuperSign oracle query, and $Succ_{A_{II}}$ is negligible.

B. SECURITY ANALYSIS

This section introduces the security analysis of the proposed mobile payment mechanism. Based on the difficulty of solving the ECDLP, we prove that our proposed payment scheme is secure against the super Type I and super Type II adversaries, respectively. That is, the certificateless signature deployed in our proposed payment scheme is existentially unforgeable against a super type adversary under the difficulty of solving the ECDLP.

Theorem 1: If there exists a $(t, q_{CU}, q_{PKR}, q_{SVE}, q_{PKE}, q_{SS})$ super Type I adversary A_I, which is able to submit additional q_H queries to random oracles *Hash* and win game 1 with probability $Succ_{SA_1}$, then there will be another algorithm B which is able to solve a random instance of the ECDLP in polynomial time with a success probability $Succ_B \geq \frac{1}{q_H}\left(1 - \frac{1}{q_H}\right)^{q_{PKE}} Succ_{A_I}$.

Proof: Assume that there exists a super type I adversary A_I which is able to break our proposed transaction scheme with a non-negligible probability $Succ_{A_I}$. The goal in this proof is to utilize A_I to construct a polynomial-time algorithm B solving the ECDLP. That is, given a random instance $(P, Q = a \cdot P)$ of the ECDLP, we would like to derive the secret a.

First, in the system initialization phase, B chooses a challenged identity ID_π in game 1. Then, B sets $Q = R_i$ and sends $params = \{G_1, G_2, q, e, P, PK_{TTP}, H_1, H_2, H_3, e(P, P)\}$ to A_I. Then, B can simulate the oracle queries of A_I as follows:

- ➢ *Hash* query: At any time A_I can access *Hash* query, which is simulated as the random oracle. That is, B maintains a list, L_H, of tuples $< ID_j, R_j, PK_{TTP}, h_j, k_j, NSD, EPD >$. If the query ID_j is already in the list L_H, then B responds with h_j (or k_j) to A_I. Otherwise, B chooses a random number $h_j \in Z_p^*$ (or $k_j \in Z_p^*$), returns h_j (or k_j) to A_I, and adds $< ID_j, R_j, PK_{TTP}, h_j, k_j, NSD, EPD >$ to L_H.
- ➢ CreateUser: At any time, A_I can request to create the user ID_j. Once it receives a query with ID_j, B first checks the other list L and then, if it is required, creates and adds a tuple into the list L based on the following two conditions. After that, B adds $< ID_j, \left(s_j, R_j\right), x_j, PK_{ID_j} >$ to the list L.
  1. If $ID_j \neq ID_\pi$, B chooses $b_j \in Z_p^*$ and $\left(s_j, R_j\right) \in Z_p^*$, and sets $PK_{ID_j} = b_j \cdot P$ and $x_j = b_j$.
  2. If $ID_j = ID_\pi$, B chooses a value of $PK_{ID_\pi} \in Z_p^*$, and sets $x_\pi = \perp$ and $\left(s_j, R_j\right) = \perp$.
- ➢ PrivateKeyExtract: At any time A_I can request the private key $\left(s_j, R_j\right)$ of the user ID_j which has been created. Once it receives a query with ID_j, B checks the list L:
  1. If $\left(s_j, R_j\right) = \perp$, B terminates the simulation.
  2. If $\left(s_j, R_j\right) \neq \perp$, B return $\left(s_j, R_j\right)$.
- ➢ PublicKeyReplace: At any time A_I can request to replace the user ID_j's public key with PK'_{ID_j} chosen by A_I. Once it receives a query with ID_j, B updates the list L by replacing the existing tuple with $< ID_j, \left(s_j, R_j\right), x_j, PK'_{ID_j} >$.
- ➢ SecretValueExtract: At any time A_I can request the secret value of the existing user ID_j. Once it receives a query with ID_j, B checks the list L:
  1. If $x_{ID_j} = \perp$, B terminates the simulation.
  2. If $x_{ID_j} \neq \perp$, B return x_{ID_j}.
- ➢ SuperSign: At any time A_I can request a SuperSign query with (ID_t, m_t). Once it receives a query with ID_j, B looks for $< ID_j, R_j, PK_{TTP}, h_j, k_j, NSD, EPD >$ and $< ID_j, \left(s_j, R_j\right), x_j, PK_{ID_j} >$ in the lists L_H and L, respectively. Next, B generates a random number $a_j, b_j \in Z_n^*$, and computes $\sigma_{j\_1} = a_j^{-1} \cdot P$ and $\sigma_{j\_2} = b_j^{-1} \cdot P$. After that, B returns σ_{j_1} and σ_{j_2} to A_I.

Finally, A_I outputs a forged but valid signature $(ID_j, m_j, \sigma_{j\_1}, \sigma_{j\_2})$. If $ID_j \neq ID_\pi$, B stops the simulation. Otherwise, B looks for $< ID_j, R_j, PK_{TTP}, h_j, k_j, NSD, EPD >$ and $< ID_j, \left(s_j, R_j\right), x_j, PK_{ID_j} >$ in the lists L_H and L, respectively. According to the forking lemma presented in [9], if we have the polynomial replay of B with the same random type and different choices of hash oracle, A_I is able to output another signature. Finally, we will have two valid signatures $\sigma_{j\_1}^{(j)}$ and $\sigma_{j\_2}^{(j)}$, where $j = 1, 2$. The two verification equations corresponding to $\sigma_{j\_1}^{(j)}$ and $\sigma_{j\_2}^{(j)}$ are “$R_i^{(j)} + h_j \cdot PK_{TTP}$” and “$k_j \cdot \left(R_i^{(j)} + h_j \cdot PK_{TTP}\right) + PK_j$”, respectively. With the known values h_j, k_j, PK_j and PK_{TTP}, it is possible to derive r_i from the above two linear verification equations. It then outputs $a = r_i$ as the solution of the random instance, i.e. $(P, Q = a \cdot P)$, of the ECDLP. So far, we have demonstrated the success of breaking the given instance of the ECDLP via B. Next, we derive B's success probability, i.e. $Succ_B$, of winning game 1.

E_1: B does not abort in all the queries of PrivateKeyExtract.

E_2: A_I is able to forge a valid signature $(ID_j, m_j, \sigma_{j\_1}, \sigma_{j\_2})$.

E_3: The output $(ID_j, m_j, \sigma_{j\_1}, \sigma_{j\_2})$ satisfies $ID_t = ID_\pi$.

The probabilities of $\Pr[E_1]$, $\Pr[E_2|E_1]$ and $\Pr[E_3|E_1 \wedge E_2]$ are presented. That is, $\Pr[E_1] \geq \left(1 - \frac{1}{q_H}\right)^{q_{PKE}}$, $\Pr[E_2|E_1] \geq Succ_{A_I}$ and $\Pr[E_3|E_1 \wedge E_2] \geq \frac{1}{q_H}$, where q_H and q_{PKE} are the numbers of *Hash* queries and PrivateKeyExtract queries. Then, the probability of B solving the given instance of the ECDLP is

$$\begin{aligned} Succ_B &= \Pr[E_1 \wedge E_2 \wedge E_3] \\ &= \Pr[E_1]\Pr[E_2|E_1]\Pr[E_3|E_1 \wedge E_2] \\ &\geq \frac{1}{q_H}\left(1 - \frac{1}{q_H}\right)^{q_{PKE}} Succ_{A_I} \end{aligned}$$

In conclusion, B is able to solve the ECDLP with a non-negligible probability $Succ_B$ if $Succ_{A_I}$ is non-negligible. This contradicts the hardness of the ECDLP.

Theorem 2: If there is a $(t, q_{CU}, q_{PKR}, q_{SVE}, q_{SS})$ super Type II adversaryA_{II} which can submit additional q_H queries to random oracles and win game 2 with probability $Succ_{A_{II}}$, then there exists another algorithm B which can solve a random instance of the ECDLP in polynomial time with a success probability $Succ_B \geq \frac{1}{q_H}\left(1 - \frac{1}{q_H}\right)^{q_{SVE}} Succ_{A_{II}}$

Proof: Assume that there exists a super type II adversary A_I which is able to break our proposed payment scheme with a non-negligible probability $Succ_{A_{II}}$. Then, we would like to establish a polynomial-time algorithm B exploiting A_{II} to solve the ECDLP. That is, given a random instance $(P, Q = a \cdot P)$ of the ECDLP, B can derive the secret a via A_{II}. Similarly, at the system initialization phase, B determines a challenged identity ID_π in game 2, Then, B sets $Q = PK_i$ and sends $params = \{G_1, G_2, q, e, P, PK_{TTP}, H_1, H_2, H_3, e(P,P)\}$ to A_{II}. Meanwhile, B maintains the lists, i.e. L_H and L. Then, B answers *Hash*, CreateUser, PublicKeyReplace, SecretValueExtract, and SuperSign as the Type I adversary does in Theorem 1. Finally, A_{II} outputs a forged but valid signature $(ID_j, m_j, \sigma_{j\_1}, \sigma_{j\_2})$. If $ID_j \neq ID_\pi, B$ stops the simulation. Otherwise, B looks for $< ID_j, R_j, PK_{TTP}, h_j, k_j, NSD, EPD >$ and $< ID_j, (s_j, R_j), x_j, PK_{ID_j} >$ in the lists L_H and L, respectively. Based on the forking lemma [9], we will eventually have two valid signatures, i.e.$\sigma_{j\_1}^{(j)}$ and $\sigma_{j\_2}^{(j)}$, where $j = 1, 2$. The two verification equations corresponding to $\sigma_{j\_1}^{(j)}$ and $\sigma_{j\_2}^{(j)}$ are "$R_i^{(j)} + h_j \cdot PK_{TTP}$" and "$k_j \cdot \left(R_i^{(j)} + h_j \cdot PK_{TTP}\right) + PK_i^{(j)}$", respectively. With these two linear and independent equations, B can derive the two unknown values r_i and x_i, and outputs x_i as the solution of the random instance $(P, Q = x_i \cdot P)$ of the ECDLP. After that, we analyze $B's$ success probability $Succ_B$ of winning game 2.

E_1: B does not abort in all the queries of SecretValueExtract.

E_2: A_{II} can forge a valid signature $(ID_j, m_j, \sigma_{j\_1}, \sigma_{j\_2})$.

E_3: The output $(ID_j, m_j, \sigma_{j\_1}, \sigma_{j\_2})$ satisfies $ID_j = ID_\pi$.

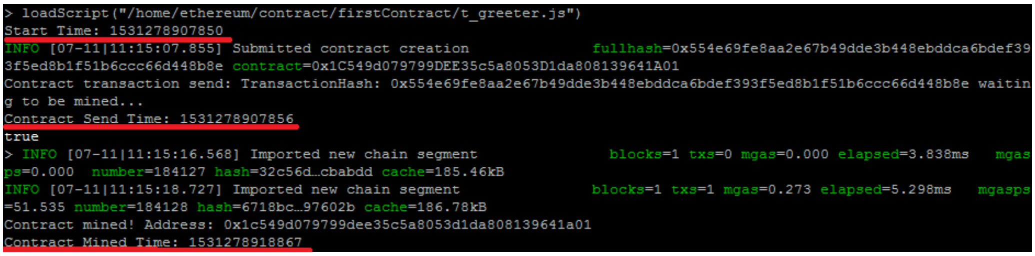
The probabilities of $\Pr[E_1] \geq \left(1 - \frac{1}{q_H}\right)^{q_{SVE}}$, $\Pr[E_2|E_1] \geq Succ_{A_{II}}$ and $\Pr[E_3|E_1 \wedge E_2] \geq \frac{1}{q_H}$, where q_H and q_{SVE} are the numbers of the *Hash* query and the SecretValueExtract query, respectively. Then, the probability of$B$ solving the given instance of the ECDLP is

$$\begin{aligned} Succ_B &= \Pr[E_1 \wedge E_2 \wedge E_3] \\ &= \Pr[E_1]\Pr[E_2|E_1]\Pr[E_3|E_1 \wedge E_2] \\ &\geq \frac{1}{q_H}\left(1 - \frac{1}{q_H}\right)^{q_{SVE}} Succ_{A_{II}} \end{aligned}$$

C. DISCUSSIONS ON SECURITY REQUIREMENTS

Sections IV (a) and (b) demonstrate the robustness of our proposed transaction scheme, which is secure against super Type I and II adversaries. In this subsection, we further discuss the achieved security requirements identified in Section I.

- During the transaction process of our proposed mobile payment scheme, six message flows, namely*NSD*, (s_i, R_i, σ_{i_1}), $(ID_i, TID, r_1, EPD, R_i, \sigma_{i\_2})$, $(T_i, Q_i, Hash_i, Cipher_1)$, $(TID_i, r_1, EPD, R_i, \sigma_{i\_1}, \sigma_{i\_2}, result)$ and (r_i, σ_{i_3}) are transmitted through unsecure network channels. Among them, *NSD* consists of non-sensitive data, and (s_i, R_i, σ_{i_1}) is the verification message for *NSD*. The sensitive part, i.e. *SD*, of the payment data *PD* is protected by $H_3(r_1 \cdot x_i \cdot PK_M)$ in the equation of *EPD*. The other four messages, namely $(ID_i, TID, r_1, EPD, R_i, \sigma_{i\_2})$, $(T_i, Q_i, Hash_i, Cipher_1)$, $(TID_i, r_1, EPD, R_i, \sigma_{i\_1}, \sigma_{i\_2}, result)$ and (r_i, σ_{i_3}) contain either verification messages, i.e. $(R_i, \sigma_{i\_2}, T_i, \sigma_{i\_3})$, operated with bilinear pairing and ECC scalar multiplication, or encrypted messages, i.e. $(Q_i, Hash_i, Cipher_1)$, well-protected by the robust one-way hash function. Note that $(ID_i, TID, r_1, r_i, result)$ are non-sensitive data also. Therefore, based on our design, the data confidentiality is naturely embedded in our proposed mobile payment scheme.
- In our proposed payment scheme, *TTP* will produce a signature message, i.e. (s_i, R_i, σ_{i_1}) constructed by bilinear pairing and ECC scalar multiplication, as a proof of the transaction data, i.e. $NSD = (TID, AM, PIs)$, launched by the user (or mobile client). Based on the proof (s_i, R_i, σ_{i_1}) provided by *TTP*, the user creates another signature σ_{i_2} associated with the encrypted payment data *EPD* which will be verified at the merchant side. Thus, the non-repudiation property holds at the user side. In addition, the mobile payment platform establishes a message $(T_i, Q_i, Hash_i, Cipher_1)$ as a proof for the existence of this transaction. Finally, the merchant is able to send all of the corresponding transaction data, i.e. $Tran_i = (TID_i, r_1, EPD, R_i, \sigma_{i\_1}, \sigma_{i\_2}, result)$, to *TTP* and asks for a signature σ_{i_3} as his/her proof for this transaction. This design ensures that the merchant cannot deny the existence of this transaction when a dispute



```
> loadScript("/home/ethereum/contract/firstContract/t_greeter.js")
Start Time: 1531278907850
INFO [07-11|11:15:07.855] Submitted contract creation              fullhash=0x554e69fe8aa2e67b49dde3b448ebddca6bdef39
3f5ed8b1f51b6ccc66d448b8e contract=0x1C549d079799DEE35c5a8053D1da808139641A01
Contract transaction send: TransactionHash: 0x554e69fe8aa2e67b49dde3b448ebddca6bdef393f5ed8b1f51b6ccc66d448b8e waitin
g to be mined...
Contract Send Time: 1531278907856
true
> INFO [07-11|11:15:16.568] Imported new chain segment               blocks=1 txs=0 mgas=0.000 elapsed=3.838ms   mgas
ps=0.000  number=184127 hash=32c56d…cbabdd cache=185.46kB
INFO [07-11|11:15:18.727] Imported new chain segment               blocks=1 txs=1 mgas=0.273 elapsed=5.298ms   mgasps
=51.535 number=184128 hash=6718bc…97602b cache=186.78kB
Contract mined! Address: 0x1c549d079799dee35c5a8053d1da808139641a01
Contract Mined Time: 1531278918867
```

FIGURE 7. The execution time of uploading a smart contract containing a transaction log to our Blockchain-based repository.

arises. It is obvious that our proposed payment scheme possesses the transaction non-repudiation property.

- Through the design of the transmitted messages, NSD, (s_i, R_i, σ_{i_1}), $(ID_i, TID, r_1, EPD, R_i, \sigma_{i\_2})$, $(T_i, Q_i, Hash_i, Cipher_1)$, $(TID_i, r_1, EPD, R_i, \sigma_{i\_1}, \sigma_{i\_2}, result)$ and (r_i, σ_{i_3}), we can see that (ID_i, ID_{MPP}, PK_M) are involved with all of the transmitted messages. Under the protection of the robust one-way hash functions, it is hard for malicious adversaries to counterfeit valid and legal transaction data based on (ID_i, ID_{MPP}, PK_M). This design prevents our proposed scheme from being attacked by man-in-the-middle (MITM) attack and impersonation attack. On the other hand, at each new session our proposed payment scheme chooses three random numbers, i.e. (r_1, t_i, r_i), to be incorporated with all of these messages. These randomly selected numbers make it so the transmitted messages cannot be used from session to session. Hence, our payment scheme is secure against replay attack and DoS attack.

TABLE 1. The experimental environment.

	Platform
Hardware	Raspberry PI 3 with 1GHz Quad-Core ARM Cortex-A53 64Bit Processor, 1GB DDR3 RAM, and SanDisk 8GB Class 10 SD Card
Software	-Operating system: Debian 8 (Raspbian 2016/10) -Programming language: Oracle Java 8 for ARM -Integrated development platform: Eclipse 3.8

D. PERFORMANCE EVALUATION FOR THE PROPOSED MOBILE PAYMENT SCHEME AT THE USER SIDE (WITH MOBILE DEVICES)

To evaluate the computation efficiency of our proposed mobile payment scheme, we implement the critical crypto-components adopted in our scheme on a common testbed, i.e. the Raspberry PI 3 model B, simulated as a mobile device held and operated by the user. Because a computation bottleneck always appears at the mobile device instead of at computation-powerful entities, such as mobile payment platforms or merchants, it thus requires a performance evaluation on the mobile device at the user side. Table 1 presents the environment of our implementation in which a Raspberry PI 3 is simulated as a mobile device at the user side. All of the adopted crypto-components in our experiments are programmed with Oracle Java 8 and Eclipse 3.8. In our proposed mobile payment scheme, the mobile device needs to perform the hash function 3 times, 1 bilinear pairing operation, ECC point multiplication (with a 384-bit prime n) 5 times, and ECC point addition (with a 384-bit prime n) 2 times, and must generate a random number. The total computation cost at the user side is around 1.11 seconds, as shown in Table 2. It is believed that our proposed payment scheme is practical for mobile devices (or even IoT (Internet of Things)-objects). Furthermore, we summarize a comprehensive comparison between our proposed scheme and existing mobile payment methods in terms of method type, highlights and evaluation (as shown in Table 3).

TABLE 2. Computation cost of our proposed mobile payment scheme at the user side (with a mobile device).

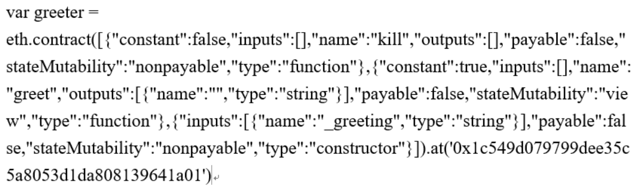
Process	Times	Cost/per	Total cost
Random number generator (96bit)	1	0.005 sec	0.005 sec
Hash function (SHA-512)	3	0.095 sec	0.285 sec
ECC Pairing	1	0.58 sec	0.58 sec
ECC point multiplication	5	0.04 sec	0.2 sec
ECC point addition	2	0.02 sec	0.04 sec
The proposed scheme			1.11 sec

E. PERFORMANC EVALUATION FOR A TRANSACTION LOG BEING UPLOADED TO A BLOCKCHAIN-BASED REPOSITORY

To evaluate the performance of the procedures related to our proposed transaction repository, we implement a private Blockchain network based on Ethereum [17] with the following setup. There are a miner and two nodes in this private Blockchain network. In order to have flexibility in terms of both hardware and software resources, miner and both nodes are VMs on different machines, respectively. The miner is allocated two Intel Core i5 6500 3.2Ghz processors, 3GB of DDR4 RAM and 127GB of hard drive allowance. Both nodes share the same configuration as the miner, but each has only one Intel Core i5 6500 3.2Ghz processor allocated to it. We connected all these participants under an 1 Gbps local network. The adopted program languages are Node js and Solidity 0.4.20. The computation time is measured as the major metric for the performance evaluation of our

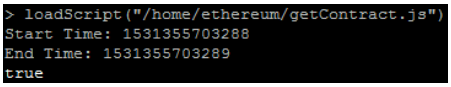
TABLE 3. Comparison of our proposed scheme and existing methods.

Study	Method Type	Highlight	Evaluation
Our scheme	System-Level	Deploy refined certificateless signature and bilinear pairing crypto-primitives as security module	Simulation on Raspberry PI 3
TrustPAY [16]	Hardware-Level	Adopt ARM TrustZone technology with OpenSSL cryptography	Simulation on ARM Cortex-A9 MPCore
Urien [12], Urien & Aghina [13-14]	Hardware-Level	Combine EMV protocol, Host Card Emulation, OpenMobileAPI, RACS protocol and TLS-SIM API to make a mobile payment on the cloud	Simulation on Android Phone supporting NFC and HCE.
Chen & Zheng [2]	Hardware-Level	A circuit design for low power 2K/4K rate RCC (range-controlled communication) chip for mobile payment	Not applicable, only circuit design concept
Park & Lee [8]	System-Level	A secure authentication scheme integrated with Signature Record Type Definition and KS X 6928 standard.	Not applicable, No simulation
Yang [15]	System-Level	Apply Identity-Based Cryptography and one-time key encryption mechanism for mobile payment	Simulation on WIS08SD548E without results



```
var greeter =
eth.contract([{"constant":false,"inputs":[],"name":"kill","outputs":[],"payable":false,"
stateMutability":"nonpayable","type":"function"},{"constant":true,"inputs":[],"name":
"greet","outputs":[{"name":"","type":"string"}],"payable":false,"stateMutability":"vie
w","type":"function"},{"inputs":[{"name":"_greeting","type":"string"}],"payable":fal
se,"stateMutability":"nonpayable","type":"constructor"}]).at('0x1c549d079799dee35c
5a8053d1da808139641a01')
```

FIGURE 8. **A query command for a smart contract containing a transaction log.**



```
> loadScript("/home/ethereum/getContract.js")
Start Time: 1531355703288
End Time: 1531355703289
true
```

FIGURE 9. **The execution time of performing a query command for a smart contract containing a transaction log from our Blockchain-Based repository.**

proposed Blockchain-based transaction repository, based on condition 1 and condition 2.

- Condition 1 models the processes whereby the system sends a smart contract containing a transaction log $(ri, \sigma_{i\_3}, Tran_i)$ to the private Blockchain network we built, and the miner then confirms that the transaction log has been successfully stored in the existing two nodes in the Blockchain network. Based on our experiment results (as shown in Figure 7), it requires 6 ms for a node to send a 2.1 KB (kilobyte) smart contract with our transaction log to the Blockchain network. After receiving the smart contract, the miner requires 11.011 seconds to complete the mining procedure. In brief, we get an execution time, i.e. 11.011 seconds, as the computation cost of condition 1, which is treated as the cost of uploading a transaction log to our proposed Blockchain-based repository.
- Condition 2 models the procedures whereby an interested party, e.g., a node, launches a *Query* command, as shown in Figure 8, for requesting a smart contract with a transaction log $(ri, \sigma_{i\_3}, Tran_i)$. The experimental results (shown in Figure 9) present an execution time, i.e. 1 ms, for condition 2.

V. CONCLUSION & FUTURE WORK

In this paper, we have presented the design of a robust transaction scheme for mobile payments. Certificateless signature and bilinear pairing crypto-primitives are elegantly integrated into our proposed transaction scheme to guarantee security robustness while preserving computation efficiency. To investigate its practicability, we implemented the proposed transaction scheme on a Raspberry PI 3 platform simulating a common mobile device. A user-acceptable computation cost, i.e. 1.11 seconds, for a regular transaction process performed at the mobile device (i.e. operated by the user) is obtained. In addition, we implemented a smart contract-based transaction repository which is based on a private Blockchain network with Ethereum. In brief, we believe that the solid system robustness and efficient computation makes our proposed mobile payment scheme one of the most promising candidates for the next generation of mobile payment technologies. Three future works are suggested. First of all, it would be interesting to investigate the possibility of further refinement of existing crypto-technologies to fulfill the specific requirements of mobile payments. Room still exists for security enhancement of mobile devices at the end-user side during mobile payments, in terms of hardware, software and underlying communication architecture. In addition, intergrating post-quantum cryptography techniques, for example as in [18] and [19], may be another good consideration as a security protection mechanism for mobile payment. Secondly, it would be promising to research the issues of how to further integrate the Blockchain technology into transaction procedures involved in mobile payments. This would more fully harness the potential of the Blockchain technology and significantly benefit the mobile payment system from a security standpoint. Thirdly, auditing is of importance for payment transaction. Therefore, integrating a full-fledged auditing mechanism into a mobile payment scheme will be a promising development direction in the future.

REFERENCES

[1] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur. (ASIACRYPT)*, Taipei, Taiwan, 2003, pp. 452–473.

[2] C. Yicheng and Z. Zhaoxia, "A low-power 2K/4K range-controlled communication chip design for mobile payment," in *Proc. 7th IEEE Int. Nanoelectron. Conf. (INEC)*, Chengdu, China, May 2016, pp. 1–2.

[3] V. Daza, R. Di Pietro, F. Lombardi, and M. Signorini, "FRoDO: Fraud resilient device for off-line micro-payments," *IEEE Trans. Dependable Secure Comput.*, vol. 13, no. 2, pp. 296–311, Mar./Apr. 2016.

[4] R. K. Garg and N. K. Garg, "Developing secured biometric payments model using tokenization," in *Proc. Int. Conf. Soft Comput. Techn. Implementations (ICSCTI)*, Faridabad, India, Oct. 2015, pp. 110–112.

[5] X. Huang, Y. Mu, W. Susilo, D. S. Wong, and W. Wu, "Certificateless signatures: New schemes and security models," *Comput. J.*, vol. 55, no. 4, pp. 457–474, 2012.

[6] G. Jetsiktat, S. Panthuwadeethorn, and S. Phimoltares "Enhancing user authentication of online credit card payment using face image comparison with MPEG7-edge histogram descriptor," in *Proc. Int. Conf. Intell. Inform. Biomed. Sci. (ICIIBMS)*, Okinawa, Japan, Nov. 2015, pp. 67–74.

[7] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. Accessed: Mar. 31, 2018. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[8] S.-W. Park and I.-Y. Lee, "Transaction authentication scheme based on enhanced signature RTD for NFC payment service environments," in *Proc. Int. Conf. Platform Technol. Service (PlatCon)*, Jeju, South Korea, Feb. 2016, pp. 1–4.

[9] D. Pointcheval and J. Stern, "Security proofs for signature schemes," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn. (EUROCRYPT)*, Saragossa, Spain, 1996, pp. 387–398.

[10] R. L. Rivest and A. Shamir, "Payword and micromint: Two simple micropayment schemes," *Int. Workshop Secur. Protocols*, Cambridge, U.K., 1996, pp. 69–87.

[11] I. Turk and A. Cosar, "An open, NFC enabler independent Mobile payment and identification method: NFC feature box," in *Proc. IEEE 17th Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Coimbra, Portugal, Jun. 2016, pp. 1–3.

[12] P. Urien, "Innovative mobile payments in the cloud for connected citizen: The MobiSIM project," in *Proc. 18th Medit. Electrotechn. Conf. (MELECON)*, Limassol, Cyprus, Apr. 2016, pp. 1–6.

[13] P. Urien and X. Aghina, "Secure mobile payments based on cloud services: Concepts and experiments," in *Proc. IEEE 2nd Int. Conf. Big Data Secur. Cloud, IEEE Int. Conf. High Perform. Smart Comput., IEEE Int. Conf. Intell. Data Secur.*, New York, NY, USA, Apr. 2016, pp. 333–338.

[14] P. Urien and X. Aghina, "The SIMulation project: Demonstrating mobile payments based on cloud services," in *Proc. IEEE 17th Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Coimbra, Portugal, Jun. 2016, pp. 1–3.

[15] Y. Rui-xia, "Design of secure mobile payment system based on IBC," in *Proc. 10th Int. Conf. Broadband Wireless Comput., Commun. Appl. (BWCCA)*, Krakow, Poland, Nov. 2015, pp. 422–425.

[16] X. Zheng, L. Yang, J. Ma, G. Shi, and D. Meng, "TrustPay: Trusted mobile payment on security enhanced ARM TrustZone platforms," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Messina, Italy, Jun. 2016, pp. 456–462.

[17] *Ethereum: Blockchain APP Platform*. Accessed: Jul. 5, 2018. [Online]. Available: https://ethereum.org/

[18] D. Xie, H. Peng, L. Li, and Y. Yang, "Efficient post-quantum secure network coding signatures in the standard model," *KSII Trans. Internet Inf. Syst.*, vol. 10, no. 5, pp. 2427–2445, 2016.

[19] D. Xie, H. P. Peng, L. Li, and Y. Yang, "Short lattice signatures with constant-size public keys," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 5490–5501, 2016.

[20] E. U. A. F. Network and I. Security. (2016). *Security of Mobile Payments and Digital Wallets*. Accessed: Aug. 23, 2018. [Online]. Available: https://www.enisa.europa.eu/publications/mobile-payments-security

KUO-HUI YEH (SM'16) received the M.S. and Ph.D. degrees in information management from the National Taiwan University of Science and Technology, Taipei, Taiwan, in 2005 and 2010, respectively. He is currently an Associate Professor with the Department of Information Management, National Dong Hwa University, Hualien, Taiwan. He has authored over 90 articles in international journals and conference proceedings. His research interests include IoT security, Blockchain, mobile security, NFC/RFID security, authentication, digital signature, data privacy, and network security. He has served as a TPC member of 26 international conferences/workshops on information security. He has served as a Guest Editor for *Future Generation Computer Systems*, the *International Journal of Information Security*, the *Journal of Internet Technology* (JIT), *Sensors*, and *Cryptography*. In addition, he has participated in the organization committee of RFIDsec'12 Asia and RFIDsec'14 Asia, NSS 2016, SPCPS 2017, and DSC 2018. He is currently an Editor of the IEEE Access, JIT, the *Journal of Information Security and Applications*, *Security and Communication Networks* and *Data in Brief*.

CHUNHUA SU received the B.S. degree from the Beijing Electronic and Science Institute in 2003 and the M.S. and Ph.D. degrees in computer science from the Faculty of Engineering, Kyushu University, Japan, in 2006 and 2009, respectively. He was with the Institute for Infocomm Research, Singapore, from 2011 to 2013. From 2013 to 2017, he was an Assistant Professor with the Japan Advanced Institute of Science and Technology, Osaka University. He is currently an Associate Professor with the Division of Computer Science, The University of Aizu, Aizuwakamatsu, Japan. His research interests include cryptanalysis, cryptographic protocols, privacy-preserving technologies in data mining, and the Internet-of-Things security and privacy.

JIA-LI HOU received the M.B.A. and the Ph.D. degrees from the National Central University, Taiwan. He is currently an Associate Professor with the Department of Information Management, National Dong Hwa University, Taiwan, China. His primary research interests include information security, data mining, financial engineering, big data, business intelligence, and enterprise resource planning.

WAYNE CHIU received the B.S. degree in information management from National Dong Hwa University, Hualien, Taiwan, in 2017, where he is currently pursuing the degree. His research interests include telephony, computer networks, Internet of Things, data analysis, and service integration.

CHIEN-MING CHEN received the Ph.D. degree from National Tsing Hua University, Hsinchu, Taiwan. He is currently an Associate Professor with the Shandong University of Science and Technology, China. He has published over 40 journal papers and 50 conference papers. His current research interests include network security, mobile internet, wireless sensor network, and cryptography.

• • •