The Journal of Supercomputing (2022) 78:2503–2527
https://doi.org/10.1007/s11227-021-03830-4

A cloud-based mobile payment system using identity-based signature providing key revocation

Fatemeh Alidadi Shamsabadi[1] · **Shaghayegh Bakhtiari Chehelcheshmeh**[1]

Accepted: 19 April 2021 / Published online: 1 July 2021

Abstract
Along with the increasing expansion of wireless networks and mobile devices, security, and efficiency in mobile payment systems have become especially important. In this research, a secure and efficient mobile payment system is provided using an Identity-Based Signature (IBS). In the proposed scheme, issues related to managing digital certificates and also the key escrow problem related to identity-based cryptosystems are resolved. In the proposed system, malicious users are not only tracked but revoked from the system. The security and correctness of the proposed protocol are analyzed theoretically and also ProVerif (Protocol Verifier) automated tool used for verifying the security of the proposed scheme formally. The proposed scheme reduces the computational overhead of mobile devices by modifying system parameters and utilizing a cloud server and demonstrates an appropriate technology to communicate between mobile devices to perform payment transactions. Moreover, the proposed protocol provides more security attributes and reduces the total running time of the signature validation algorithm server-aided compared to existing similar protocols.

Keywords Mobile payment system · Identity-based signature · Key revocation · Cloud computing · ProVerif

1 Introduction

Today, mobile payments are a proper alternative to traditional payment methods. Analysts have dubbed the event "the death of cash" [1].

Mobile payment systems allow individuals or organizations to do secure financial transactions with other people through a wireless network using mobile devices.

✉ Shaghayegh Bakhtiari Chehelcheshmeh
sh.bakhtiari@iaushk.ac.ir

[1] Department of Computer, Faculty of Engineering, Shahrekord Branch, Islamic Azad University, Shahrekord, Iran

Each mobile payment system includes four principal entities: the customer, the merchant, the merchant's financial institution (as the acquirer), and the customer's financial institution (as the issuer). Also, the mobile payment system can include an additional entity called the payment gateway, which acts as an intermediate for payment-clearing purposes between the acquirer and the issuer. These entities communicate with each other using an internet access service. Communications among the issuer, the acquirer, and the payment gateway take place over the private banking networks, and the security of these communications is provided using cryptographic mechanisms [2].

Mobile payment systems based on symmetric cryptography use a shared secret key. For this reason, the robustness and efficiency of these algorithms depend on the key length used and secure transmission of keys between both sides of the transaction. Also, symmetric cryptography cannot provide non-repudiation and authentication requirements if an attacker is successful in compromising the symmetric key, then both customer and merchant get exposed and all the payment transactions encrypted with that key are exposed [3, 4].

To avoid the above issues, new generations of mobile payment systems are making extensive use of public key cryptography. These systems use a pair of keys (a public key that is published and a private key that remains hidden) for both sides. Therefore, there is no need to share the secret key between the customer and merchant before the secure connection. Also, for providing non-repudiation and authentication, these algorithms use the digital signature [4].

However, the main problem in developing mobile payment systems based on public key cryptography is the deployment and management of infrastructure to implement the authenticity of cryptographic keys. For solving this problem, the Public Key Infrastructure (PKI) is used, in which users' digital certificates are issued by a Certificate Authority (CA). However, the use and management of PKI are often considered costly; because it includes revocation, storage, distribution, and verification [5].

Fortunately, IBS systems have been introduced to solve digital certificate issues in the public key cryptosystem [6]. All user keys in these types of systems are generated based on their identity information (such as email address); for this reason, a certificate is not required to prove the authenticity of users. However, ID-based mobile payment systems have a key escrow problem; because users' private keys are issued by a third party called the Key Generation Center (KGC), KGC can find all the user's confidential information or forge the user to generate a valid signature based on his identity [5].

In addition to security issues, mobile devices used in mobile payment systems are facing many challenges in their resources such as low battery life, limited storage space, and low bandwidth. Limited resources significantly prevent the improvement of service qualities [7].

Therefore, a secure and efficient mobile payment system should address the security issues mentioned above; also, the computational overhead on mobile devices should be minimized and appropriate communication technology should be used among these devices to perform payment transactions. To achieve this, the present research study offers a mobile payment system with the ability of tracing and

revocation of malicious users through an identity-based signature scheme wherein the issues related to digital certificates are resolved. In this proposed protocol, the keys chosen by the users for transactions are used to address the key escrow problem. The ProVerif automated tool is also used to ensure the security of the proposed protocol against known active and passive attacks. In order to reduce computational overhead on mobile devices, some calculations pertinent to the time key update and signature verification are outsourced to the cloud server. The scheme also demonstrates a suitable communication technology between mobile devices for payment transactions.

The next sections of this article are as follows: In Sect. 2, earlier works are recalled. In Sect. 3, the background of the proposed scheme is illustrated. In Sect. 4, the proposed scheme is presented. In Sect. 5, the proposed protocol is compared and evaluated with other similar protocols, and in Sect. 6, a general conclusion is given.

2 Related works

So far, the issue of mobile payment security has been discussed by many authors. Here is an overview of previous works in this area. It should be noted that requirements for security, such as identification, authentication, security message protocols, messaging and data cryptography, differ for each technology generation and each payment scenario [8]. Therefore, in order to evaluate mobile payment systems, these should be considered.

One of the oldest mobile payment systems could be the proposed system by Chaum [9] in 1983, based on blind signatures. In this solution, the requester could get a valid signature on the message without showing the message content to the signer. Later, in 2003, Chang and Lai [10] presented a date-attachment blind signature scheme for e-cash. Juang [11] also introduced a pre-paid e-cash system for date attachment based on blind signature in 2007. Unfortunately, blind signature methods are weak against a chosen text attack due to the use of the RSA algorithm [12, 13].

In 2007, Bisel [14] examined the role of the Secure Sockets Layer (SSL) in cybersecurity. To this end, it outlined the strengths and limitations and weaknesses of SSL. Then, SSL and Internet Protocol security (IPsec) were compared and evaluated. Guan [15] in 2009 reviewed a Secure Electronic Transaction (SET)-based payment system and concluded that the protocol would lead to more security and efficiency in the payment transactions. In 2012, Frisby et al. [16] analyzed the security of smartphone point-of-sale systems. They showed that all payment applications using Transport Layer Security (TLS) are protected against an attacker on the network. Leu et al. [17] reviewed SSL and SET in 2015 and proposed a Secure Mobile Commerce System (SMCS) to improve these protocols.

SSL and SET use both public key and private key as a basis for securing the message exchange process. In payment transactions using SSL, it's just enough for a person to send the merchant, card number, expiration date, and other information. However, at SET, the customer and merchant must apply for the certificate so that they

can receive the electronic certification of SET and its software from the card-issuing bank. Then, they use the software to complete a transaction online. Therefore, in SET, unlike SSL, both sides of the transaction can authenticate each other. Also, the SET can protect consumer's credit data, since the merchant needs only a consumer's SET credential before it can bill the card-issuing bank.

As customers and merchants apply for a SET certificate, credit card information of the user is stored on a hard disk. Also, SET spends a lot of time for computing asymmetric keys (keys for encryption and decryption) to improve security, resulting in an unpleasant experience of mobile commerce for users. To address these issues, the SMCS proposes a credit card in which a dynamic authentication code has been replaced with credit card information so that merchants cannot know the number of credit cards and their details. To establish a secure channel is required in mutual authentication and provide a secure wireless exchange of the messages, the SMCS uses a Data Connection Core (DCC) to connect the card-issuing bank and the client in before starting a wireless connection. It also provides confidentiality property by a two-dimensional stream cipher technique. Leo et al. simulated this protocol using Java. The most important simulation approach was that SMCS has very high computational overhead due to the use of various functions such as RSA, AES (Advanced Encryption System), and HMAC (Hash-based Message Authentication Code).

In 2015, Pelaez et al. [18] presented a Person to Person Mobile payment (P2PM-pay) scheme based on mobile cash. They also designed a Wireless Public Key Infrastructure (WPKI) to perform the authentication and user certificate revocation process.

In PKI-based schemes, the validity of each certificate is generally limited to one expiration date. Ideally, certificates are expected to be used for their entire validity. Unfortunately, there are situations in which a certificate must be revoked before its expiration date; for example, if the private key of a certificate is lost or exposed. So for such cases, a mechanism is required to revoke the certificate. For this reason, mechanisms such as Certificate Revocation List (CRL) and Online Certificate Status Protocol (OCSP) were introduced to revoke the certificate before its expiration date [19].

Generally, public key infrastructure is a security solution for mobile payments that requires a lot of computing overhead [20]. Wireless PKI designed by Pelaez et al. [18] uses the Wireless Transfer Layer Security Protocol (WTLS), which to some extent avoids the problems of traditional PKI (especially computing overhead on mobile devices). The system avoids the rapid growth of the bank's database, since the expiration date is embedded into the mobile cash by partial blind signature during the withdrawal date, and the bank does not hold information about the operation. But because wireless PKI also uses certificate-based digital signatures, it imposes restrictions on users such as high bandwidth, network latency, and additional costs [2].

In [21] Isaac and Zeadally presented a payment protocol in a payment gateway centric model. All payments sent between entities of this protocol are transferred through the payment gateway. Therefore, customers do not directly communicate with the merchant and the anonymity of consumers is provided. In this protocol, symmetric cryptography is used to provide the confidentiality property, and the

Message Authentication Code (MAC) is used to ensure the message integrity. This mechanism also fulfills the requirements of cloud computing, because the payment gateway can be used as a cloud service provider for mobile payments. However, this protocol does not provide the non-repudiation requirement and affects symmetric cryptographic issues, especially the key management problem. As a result, it is not suitable for mobile payments in cloud computing.

Yang and Lin [22] in 2016, proposed a similar mechanism to improve the Isaac and Zeadally protocol [21]. In this protocol, there is a payment gateway between the customer and the merchant, and the customer uses the anonymous identity to create payment information. Hence, the merchant does not know who is buying the goods. The customer's bank also does not know what she is buying, since transaction information and payment information are protected by the one-way hash function. The customer's bank generates the digital signature as a payment proof; thus, the non-repudiation requirement is provided. The scheme uses the RSA algorithm [23] to provide confidentiality; consequently, the key management problem is solved, but the key length is very high (less performance).

Qin et al. [24] in 2017 used the certificate-less signing scheme belongs to Huang et al. [25] in their protocol; This signature scheme is based on bilinear pairings, which provide requirements of non-repudiation, unforgeability and traceable messages transmitted. Also, the signature scheme of Huang et al. solves issues relating to certificate-based digital signatures and key escrow problems. Qin et al. combined the idea of outsourcing and certificate-less signature to reduce computational overhead on mobile devices. Also, to provide anonymity, they kept confidential the identity of the customer at all stages of the transaction; in this way, they transformed the customer's real identity into two pseudo-identities using the tamper-proof device. This pseudo-identity is an ElGamal-type cipher-text in the Elliptic Curve Cryptosystem (ECC) [26] that remains secure against chosen-plaintext attacks.

In 2018, Liao et al. [27] reviewed the mobile payment protocol provided by Qin et al. [24]. Liao et al. described a colluding attack in Qin's protocol which the customer and the untrusted cloud server can collude with each other and cheat the merchant. Hence, Liao et al. used hard computational assumptions to verify an outsourced signature for eliminating colluding attacks. Liao et al. also pointed out that the structure of Qin's protocol and its structure was unreasonable since, in the signing algorithm of their protocol, a point over an elliptic curve is added with an element of a set, which is not to be defined. To solve this problem, Liao et al. defined two points over an elliptic curve for adding operation in calculating the signature.

The evaluation of the newest mobile payment systems is given in the Table 1. According to reviews, just in schemes [24, 27] has been solved the problems of managing digital certificates, and key escrow. However, even in these mobile payment systems, there is no possibility of revocation malicious users, and the security of these protocols in a formal manner is not guaranteed against known active and passive attacks. These protocols do not demonstrate the appropriate technology to communicate between mobile devices. Also, the executing time of the signature validation algorithms server-aided in these protocols is high.

3 The background of proposed scheme

Mobile payment systems should provide the security properties unforgeability, non-repudiation, anonymity, traceability, and revocation of malicious users. They must be resistant to known active and passive attacks. Also, these systems must use a reasonable solution to communicate between mobile devices and minimize computational overhead.

For resolving the above issues, the proposed protocol uses an unforgeable IBS based on bilinear pairing. Because, among the various types of identity-based systems (such as quadratic residues, discrete logarithms, bilinear pairing, and lattice), systems that are based on bilinear pairing have lower computational overhead and more efficient [28]. This protocol also addresses the issues related to digital certificates and provides non-repudiation and traceability properties. In this scheme to resolve the problem of the key escrow, users choose the keys (which are not known by the KGC) for transactions. Additionally, the proposed scheme to protect users' privacy uses the pseudo-identity method, and to provide key revocation uses a list contains the pseudo-identity of malicious users. The proposed system reduces the computational overhead of mobile devices by utilizing a cloud server and modifying system parameters and it uses the appropriate technology. The ProVerif automated tool is also used to verify the security of the proposed protocol against known attacks formally.

3.1 The architecture of the proposed scheme

In this section, the architecture of the proposed protocol is presented. The main drivers of mobile devices in this scheme that enables convenient and secure mobile commerce services are the following [29, 30]:

1. Mobile wallet application (such as Google Wallet) which is a program or service that allows users to store and control their shopping information, like logins, passwords, shipping address, and credit card details, in one central place
2. Mobile payment application for buying and billing
3. Internet access with 3G/4G broadband
4. Low computational overhead (high performance)
5. Multi-media contents from merchants to stir purchasing desire on the move, extending from mobile commerce players (such as eBay, Amazon, Google, and etc.)

In the proposed system by using near-field communication (NFC), the mobile device is able to provide the customer with information about products that she is currently observing. For example, a mobile device can identify product features including its ingredients content, its price, and its expiry date [31].

Generally, to communicate between mobile devices use well-known communication protocols that currently, NFC technologies are the leading form of mobile

Table 1 The evaluation of the newest mobile payment systems

Protocols		Properties
Isaac and Zeadally [21]	Security advantages	Anonymity, confidentiality, and integrity
	Challenges	Suffering from problems of symmetric cryptography (especially key management, and failing to provide non-repudiation property)
Pelaez et al. [18]	Security advantages	Confidentiality, integrity, authentication, non-repudiation, and certificate revocation
	Challenges	Exchanging additional information, increasing latency, and high cost (due to the use of WPKI and digital signature based on certificate)
Leu et al. [17]	Security advantages	Confidentiality, integrity, authentication, and non-repudiation
	Challenges	Very high computational overhead (due to the use various functions such as RSA, AES, and HMAC)
Yang and Lin [22]	Security advantages	Anonymity, confidentiality, integrity, and non-repudiation
	Challenges	Low efficiency due to the long key length used in the RSA algorithm
Qin et al. [24]	Security advantages	Unforgeability, anonymity, traceability, and non-repudiation
	Challenges	High computational overhead, the likelihood of a colluding attack, and no key revocation
Liao et al. [27]	Security advantages	Unforgeability, anonymity, traceability, and non-repudiation
	Challenges	High computational overhead and no key revocation

payment methods. Many organizations using mobile payment services are adopting NFC systems to replace credit cards. In addition, communication security has been enhanced by using standard techniques to enable NFC services. Google, Apple, and Samsung in order to replace credit card transactions have released new mobile payment services that most of these services operate based on the NFC standard [32].

The NFC technology allows peer-to-peer communication between NFC-enabled devices like NFC phone and NFC Point-of-Sales (PoS). NFC-enabled devices and contactless PoS terminals execute payment transactions at the range of a few centimeters using communication protocols that are based on Radio-Frequency IDentification (RFID) standards [33]. Figure 1 shows a type of mobile device that integrates with the NFC technology.

As shown in Fig. 1, the NFC-enabled device is usually composed of various integrated circuits, an NFC interface, and Secure Element (SE). The communications between NFC devices are enabled over the NFC interface. This interface is composed of a contactless front-end, an RF antenna, and an NFC controller that RF antenna is responsible for the transmission and reception of radio signals [33, 34].

The NFC-enabled device incorporates SE to securely store confidential information (such as user account information) and to connect to the NFC controller to perform secure proximity transactions with external NFC devices [33].

However, there may be cloud-based solutions relying on Host Card Emulation (HCE). HCE removes the need for SE but it results in challenges of security for storing the payment information. Thus, HCE can be combined with other solutions giving a higher level of security (such as SE) [29].

In the proposed protocol using the hybrid model of HCE and SE would provide a highly secure environment. Also, the technology of the proposed scheme based on a

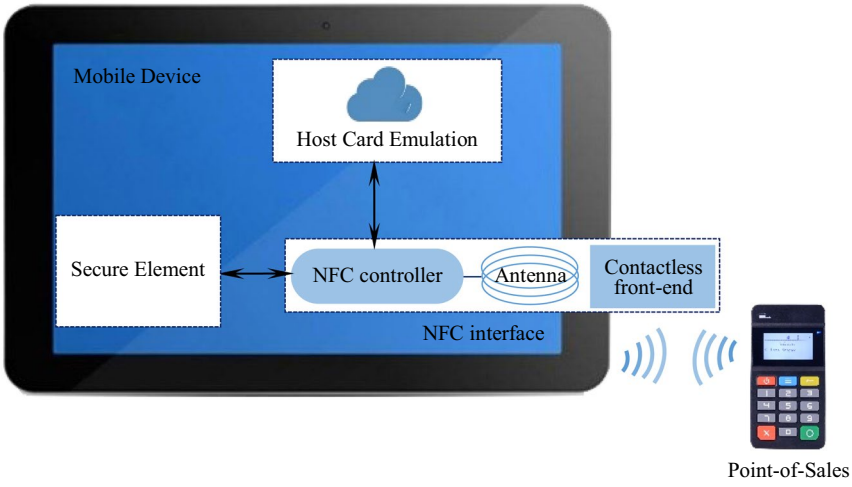

Fig. 1 The architecture of the NFC-enabled device with hybrid model SE and HCE

combination of the microSD card and NFC antenna that allows the mobile device to communicate with the contactless readers (such as PoS).

Therefore, the SE can be located on the microSD card, while the mobile device takes care of the physical NFC functionality. This solution with the SE stored on the microSD card does not depend on the network operator or device manufacturer, and thus may look attractive for financial institutions as it allows the bank institute that issues the card, to own the SE. Another advantage of the SD-based SE solution is to facilitate rapid application deployment [29].

In addition to the components listed above, the proposed protocol consists of the main entities of the customer (Alice), the merchant (Bob), the KGC, and the cloud server.

Cloud server can store and process huge data for users. Accordingly, the user of the cloud system does not need to spend expensive costs to increase processing capacity, storage, and battery life on his mobile device. Additionally, mobile cloud computing improves the quality of communications by upgrading bandwidth and reducing data delivery time [7, 35].

The mobile devices such as smartphones and tablets can access cloud services through wireless networks. These devices are connected via base transceiver stations satellites to wireless networks. Base transceiver stations satellites are accountable for controlling the connections and functional interfaces between wireless networks and mobile devices.

Mobile devices transmit the mobile users' requests and transactions to the base station controllers providing a wide range of wireless network services (such as authentication, authorization, and accounting). The subscribers' requests are then delivered through the Internet to a cloud.

Within the cloud, cloud controllers process the requests and provide mobile users with the corresponding cloud services relying on ubiquitous computing, virtualization, and service-oriented architecture connect to data centers and application servers.

The customer in a private cloud is accountable for guarantee security; but in a public cloud, security accountabilities are shared between the cloud service provider and customers. The customer is accountable for securing applications and data deployed on the cloud platform and cloud service [4].

The interaction of above entities and components is shown in Fig. 2. This interaction is divided into four phases of setup and key generation, time key update, payment transaction, and outsourced verification. In the following, each of these phases is described in detail.

To revoke the user key, the KGC maintains a Malicious-users List (ML). This list contains the pseudo-identity of users whose keys need to be revoked. ML is updated at specific time periods. The cloud server updates the user's time keys according to the ML received from the KGC in the time key update phase. The cloud server checks ML; if the user's pseudo-identity is in ML, the cloud server does not generate a new time key for that user. This will automatically revocation the malicious user from the system.

- **Setup and Key Generation Phase:** In this phase, KGC selects security parameters as input. Then, it generates the public parameters, the self-private key, the time's private key, and time-period list $T = (T_0, T_1, \ldots)$ as output. The KGC keeps the self-private key secretly and sends the time's private key through a secure channel for the cloud server, and publishes the public parameters for all system entities. Additionally, it considers the user's real identity and the self-private key as input; then it generates a pseudo-identity and partial private key for the user and sends it securely. In this phase, each user chooses a full private key, which is not known by the KGC.
- **Time Key Update Phase:** When the cloud server receives an update request from a user, it checks whether the user's pseudo-identity is found on the ML or not? If the user's pseudo-identity is in the ML, the cloud server does not accept the update request for that user; otherwise, the cloud server will generate a new time key for the user using the public parameters, time private key, user pseudo-identity, and current time-period.
- **Payment Transaction Phase:** In this phase, Alice performs a payment transaction with its full private key, partial private key and time key. Bob also uses the same method to signs a receipt and delivers it to Alice.
- **Outsourced Verification Phase:** In this phase, some calculations of signature verification are outsourced to the cloud server. Thus, the computation overhead on the user side significantly decreases.

For a better understanding, the symbols used in the proposed protocol are given in Table 2.

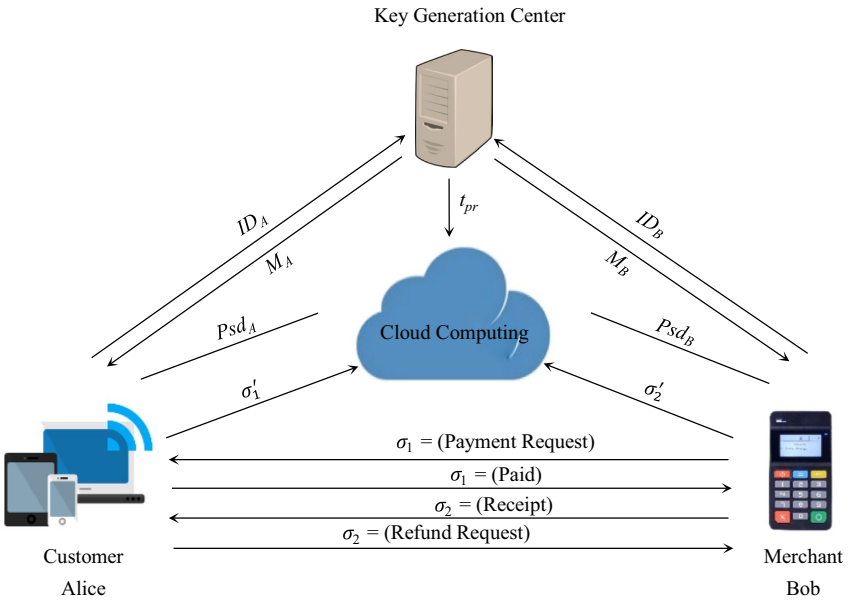

Fig. 2 The framework of the proposed protocol

3.2 Security requirements

The proposed scheme provides the following security requirements:

Unforgeability: No one should impersonate any user and thus create a fake signature.
Anonymity: The real identity of users at all stages of the payment transaction must hold hidden.
Traceability: Malicious users should be trace.
Ability to revoke a key: When a person is abdicated from a responsibility or is identified as a malicious user, that user should be revoke from the system.
Non-repudiation: None of the users should be able to deny their sent messages.

4 Proposed scheme

In this section, a new mobile payment protocol is provided using identity-based signature. The proposed protocol based on bilinear pairings and hard computational assumptions is as follows. For convenience, an additive cyclic group G_1 and a multiplicative cyclic group G_2 of the order q is considered, where q is a large prime number. Let P be a generator of group G_1. Then, $\hat{e} : G_1 \times G_1 \rightarrow G_2$ is an acceptable bilinear mapping if it satisfies the following properties:

Bilinear property For all $P, Q \in G_1$ and $w, z \in Z_q^*$ the relation $\hat{e}(wP, zQ)=\hat{e}(P, Q)^{wz}$ is established.

Non-degenerate property For all $P, Q \in G_1$ the relation $\hat{e}(P, Q) \neq 1_{G_2}$ is established.

Computable property For all $P, Q \in G_1$ exist a polynomial-time algorithm to compute $\hat{e}(P, Q) \in G_2$.

To prove the security of the proposed scheme in the random oracle model, used the following hard assumptions:

Discrete logarithm (DL) problem Given (P, xP) then the x must be computed. Here, $P \in G_1$ is a group generator and $x \in Z_q^*$ is considered as an unknown parameter.

Computational Diffie–Hellman (CDH) problem Given (P, xP, yP) then the parameter xyP must be computed. Here, $P \in G_1$ is a group generator, and $x,y \in Z_q^*$ is an unknown parameter.

In hard problems, it is assumed that there is no algorithm with a polynomial-time that can solve them with non-negligible probability.

As noted in Sect. 3.1, the proposed scheme includes the following phases.

4.1 Setup and key generation phase

To setup, KGC selects two cyclic groups G_1 and G_2 of same prime order q. It considers P as a group generator of G_1 and the mapping $\hat{e} : G_1 \times G_1 \rightarrow G_2$ as a bilinear

pairing. Also, it selects two random elements $K_{\text{pr}}, t_{\text{pr}} \in Z_{\text{q}}^*$ and three one-way hash functions H_1: $\{0,1\}^* \rightarrow G_1$, $H_2 : \{0,1\}^* \times \{0,1\}^* \rightarrow G_1$, and $\text{h} : \{0,1\}^* \times G_1 \rightarrow Z_q^*$. Then, it computes $K_{\text{pub}} = K_{\text{pr}}P$, $t_{\text{pub}} = t_{\text{pr}}P$ and $\beta = \hat{e}(P, P)$. KGC keeps its private key K_{pr} as a secret and sends time private key t_{pr} to cloud server via a secure channel. It also publishes the public parameters $P_{\text{params}} = (G_1, G_2, q, P, \hat{e}, K_{\text{pub}}, \text{t}_{\text{pub}}, \beta, H_1, H_2, h)$. Finally, the KGC and users for key generation perform the following steps:

Step 1: Alice chooses an $ID_A \in \{0,1\}^*$ as her real identity and sends ID_A to KGC. After that, KGC generates $P\text{sd}_A$ as Alice's pseudo-identity and M_A as Alice's partial private key as follows:

Pseudo-identity generation step: In this step, KGC selects $w \in Z_q^*$ randomly. Then, it computes $\text{Psd}_a = wP$ and $\text{Psd}_a^{'} = ID_A \oplus H_1(wK_{\text{pub}})$. Finally, it considers $Psd_A = (Psd_a, \text{Psd}_a^{'})$ as a pseudo-identity for Alice.

Partial private key generation step: At first, the KGC computes $Q_{P\text{sd}_A} = H_2(P\text{sd}_A)$. Then, it considers $M_A = K_{pr}Q_{P\text{sd}_A}$ as a partial private key for Alice. The KGC sends the pair (Psd_A, M_A) to Alice through a secure channel.

Step 2: In this step, Alice selects a random element $F_A \in Z_q^*$ as her full private key.

Step 3: Bob does the three steps listed above to compute pseudo-identity Psd_B, partial private key M_B, and full private key F_B.

4.2 Time key updating phase

This phase has the following steps:

Step 1: When the cloud server receives Alice's request for updates, it computes $Q_{\text{upd}_A} = H_2(\text{Psd}_A, T_i)$ and $T_{\text{upd}_A} = t_{\text{pr}}Q_{\text{upd}_A}$ base on the period T_i and then sends T_{upd_A} to Alice. Here, $P\text{sd}_A$ is the pseudo-identity of Alice.

Step 2: Bob gets T_{upd_B} like Alice using step 1.

4.3 Payment transaction phase

This phase includes the following steps:

Step 1: First of all, Bob generates a payment request $\text{Payment} = T_{\text{ID}} \| \text{Amount}_{\text{T}} \| \text{Psd}_B$ and sends it to Alice.

Step 2: When Alice receives the payment request, she uses her full private key, partial private key and time key for generating a signature as follows:

$$\alpha = \beta^{F_A},$$

$$v = h(\text{Payment}, \alpha),$$

$$U = F_{\text{A}}P + v(M_A + T_{\text{upd}_A}).$$

Finally, payment signature based on period T_i is $\sigma = (U, \alpha)$.

Step 3: When Bob receives σ, Payment, T_i and pseudo-identity of Alice, he computes $v = h(\text{Payment}, \alpha)$. Then, checks if $\hat{e}(U,P) = \alpha\hat{e}(Q_{\text{Psd}_A}, K_{\text{pub}})^v\hat{e}(Q_{\text{upd}_A}, t_{\text{pub}})^v$ holds. If it was correct, he accepts the output. Otherwise, he stops the process. The correctness of relation above as follows:

$$\begin{aligned}\hat{e}(U,P) &= \hat{e}\left(F_A P + v(M_A + T_{\text{upd}_A}), P\right)\\ &= \hat{e}\left(F_A P, P\right)\hat{e}(M_A,P)^v\hat{e}(T_{\text{upd}_A},P)^v\\ &= \hat{e}(P,P)^{F_A}\hat{e}(K_{\text{pr}}Q_{\text{Psd}_A},P)^v\hat{e}(t_{\text{pr}}Q_{\text{upd}_A},P)^v\\ &= \beta^{F_A}\hat{e}(Q_{\text{Psd}_A},K_{\text{pr}}P)^v\hat{e}(Q_{\text{upd}_A},t_{\text{pr}}P)^v\\ &= \alpha\hat{e}(Q_{\text{Psd}_A},K_{\text{pub}})^v\hat{e}(Q_{\text{upd}_A},t_{\text{pub}})^v\end{aligned}$$

Step 4: When Bob verified Alice's signature and ensured the integrity of payment information, he sets receipt information as $\text{Receipt} = T_{ID}||\text{Amount}_R||\text{Psd}_B$ and sings it as follows:

$$\alpha = \beta^{F_B},$$

$$v = h(\text{Receipt}, \alpha),$$

$$U = F_B P + v(M_B + T_{\text{upd}_B}).$$

Step 5: Alice can verify $\sigma = (U,\alpha)$ received by Bob as that Step 3 during this phase.

4.4 Outsourced verification phase

In this phase, we use the difficulty of computational assumptions to reduce the computational overhead associated with the signature verification of the cloud server and to prevent the colluding attack (details in Sect. 5.1). For this purpose, Bob and cloud server follow the steps below:

Step 1: When Bob receives σ, Payment, T_i and the pseudo-identity of Alice, he computes $v = h(\text{Payment}, \alpha)$. Then he selects two random elements $x,y \in Z_q^*$ and computes $\sigma' = xU + yP$. Finally, he sends σ', T_i and pseudo-identity of Alice to the cloud server.

Step 2: Cloud server computes β_1 and β_2 as relation follows:

$$\beta_1 = \hat{e}\left(\sigma', P\right),$$

$$\beta_2 = \hat{e}(Q_{\text{Psd}_A}, K_{\text{pub}})\hat{e}(Q_{\text{upd}_A}, t_{\text{pub}}).$$

Then, it sends β_1 and β_2 to Bob.

Step 3: Bob checks the condition $\beta_1 = \alpha^{x}\beta_2^{xv}\beta^{y}$. If the condition is true, the output is accepted by Bob. Otherwise, the process is stopped. The correctness of this condition as follows:

$$\begin{aligned}
\beta_1 &= \hat{e}\left(\sigma', P\right)\\
&= \hat{e}(xU + yP, P)\\
&= \hat{e}(xU, P)\hat{e}(yP, P)\\
&= \hat{e}\left(xF_A P + xv(M_A + T_{\mathrm{upd}_A}), P\right)\hat{e}(yP, P)\\
&= \hat{e}\left(xF_A P, P\right)\hat{e}\left(xM_A, P\right)^{v}\hat{e}(xT_{\mathrm{upd}_A}, P)^{v}\hat{e}(yP, P)\\
&= \hat{e}(xP, P)^{F_A}\hat{e}(xK_{\mathrm{pr}}Q_{\mathrm{Psd}_A}, P)^{v}\hat{e}(xt_{\mathrm{pr}}Q_{\mathrm{upd}_A}, P)^{v}\hat{e}(yP, P)\\
&= \hat{e}(P, P)^{xF_A}\hat{e}(xQ_{\mathrm{Psd}_A}, K_{\mathrm{pr}}P)^{v}\hat{e}(xQ_{\mathrm{upd}_A}, t_{\mathrm{pr}}P)^{v}\hat{e}(yP, P)\\
&= (\beta^{F_A})^{x}\hat{e}(xQ_{Psd_A}, K_{\mathrm{pub}})^{v}\hat{e}(xQ_{\mathrm{upd}_A}, t_{\mathrm{pub}})^{v}\hat{e}(yP, P)\\
&= \alpha^{x}\left(\hat{e}(Q_{\mathrm{Psd}_A}, K_{\mathrm{pub}})\hat{e}(Q_{\mathrm{upd}_A}, t_{\mathrm{pub}})\right)^{xv}\hat{e}(P, P)^{y}\\
&= \alpha^{x}\beta_2^{xv}\beta^{y}.
\end{aligned}$$

Step 4: By following the steps above in this phase, Alice can check the validity of Bob's signature.

5 The proposed scheme analysis

In this section, the proposed scheme is evaluated by security and efficiency metrics. As discussed in Sect. 2, systems that are based on identity-based signatures prevent issues related to digital certificates and public key infrastructure. For this reason, only mobile payment systems based on IBS [24, 27] are used to perform comparisons.

5.1 Security analysis

Now, the security of the proposed scheme is analyzed as follows:

- **Unforgeability:** In the proposed scheme, only legal users can generate make signatures of the payment and receipt requests; because used the unforgeable signature scheme [36] for this work.
- **Anonymity:** The identity of users is kept secret at all stages. The real identity of each user is converted into a pseudo-identity pair $(P\mathrm{sd}_i, \mathrm{Psd}_i^{'})$ for the unknown $w \in Z_q^*$. This pseudo-identity pair is an ElGamal-type ciphertext in the ECC that is resistance against chosen-plaintext attacks. Therefore, without the KGC's private key K_{pr}, there is no way to extract the real identity of users from their pseudo-identity pair.

- **Tractability:** KGC can obtain the pseudo-identity pair through payment signature or the signature of receipt. Then, it can use the self-private key to extract the user's real identity.
- **Key revocation property:** When the user is denied responsibility or the person is identified as a malicious user, KGC puts the pseudo-identity of the user in an ML and sends the ML to the cloud server. Thus, when the user sends an update request to the cloud server, the cloud server by checking the ML found that the user's pseudo-identity exist in the list; therefore, it does not accept the time key update request of the user.
- **Non-repudiation:** Bob and Alice cannot deny receipt signature and payment signature, respectively. Otherwise, KGC can be tracing and revocation them.
- **Secure against colluding attack:** If Alice colludes with the cloud server in the outsourced verification phase, Bob not accepted Alice's invalid signature; because the discrete logarithm problem is used in the structure of σ'. Therefore, the cloud server cannot obtain x and y through the relation between σ' and U. As a result, the proposed scheme remains secure against colluding attack.
- **Solving the key escrow problem:** In the proposed scheme, each user chooses a fully private key, which even the KGC is unaware of it. Then, users use their fully private keys to do their transactions. In this way, KGC's potential misconduct is prevented.

5.2 Formal security validation Using ProVerif

Here, the security of the proposed protocol is evaluated using ProVerif [37, 38]. ProVerif is an automated security tool for validating security protocols against known active and passive attacks. This tool supports a wide range of cryptographic primitives and is defined by predefined rules. Also, it can check every protocol for an unbounded message space and an unbounded number of sessions.

Implementations of the proposed scheme using ProVerif in both modes the original validation algorithm and the validation algorithm server-aided are shown in Fig. 3a–e. These implementations include three parts of the declaration, the process, and the main part. The declaration part, the KGC process, and the main part are the same in both implementations. In the declaration part, channels, constants, and variables besides cryptographic functions are delineated as constructors and equations. In the process part, the definition of processes and sub-processes are elaborated. In the main part, the initiation and termination of participating users are specified, and execution processes are kept parallel. Finally, three queries are executed in order to rectify the correctness and security of the proposed scheme.

The results of all queries (in both the original validation algorithm and the validation algorithm server-aided) are presented in Fig. 4. In these results, the correctness of the proposed scheme is substantiated, since the first two questions are executed successfully. Also, its security is confirmed due to unsuccessful query attack on session key Kij.

```
(****************************** Channels ******************************)
free CH_P: channel.              (* Public Channel *)
free CH_S: channel [private].    (* Secure Channel *)
(************************* Constants & Variables *************************)
const P: bitstring.
const tpr: bitstring.
free IDA: bitstring.
free IDB: bitstring.
free FA: bitstring [private].
free FB: bitstring [private].
free Kpr: bitstring [private].
free ML: bitstring.
free TID: bitstring.
free AmountT: bitstring.
free AmountR: bitstring.
(****************************** Constructor ******************************)
fun H1(bitstring): bitstring.
fun H2(bitstring): bitstring.
fun H3(bitstring, bitstring): bitstring.
fun H4(bitstring, bitstring, bitstring, bitstring): bitstring.
fun XOR(bitstring, bitstring): bitstring.
fun EPM(bitstring, bitstring): bitstring.
fun BL(bitstring, bitstring, bitstring): bitstring.
fun BL0(bitstring, bitstring): bitstring.
fun CON(bitstring, bitstring): bitstring.
fun CON2(bitstring, bitstring, bitstring): bitstring.
fun MUL(bitstring, bitstring): bitstring.
fun MUL1(bitstring, bitstring, bitstring): bitstring.
fun EXP(bitstring, bitstring): bitstring.
fun SUM(bitstring, bitstring): bitstring.
```

(a) The declaration part of the ProVerif code.

Fig. 3 **a** The declaration part of the ProVerif code. **b** The KGC process part of the ProVerif code. **c** The code snippet related to process part of validation algorithm original. **d** The code snippet related to process part of validation algorithm server-aided. **e** The main part of the ProVerif code

5.3 Security comparison

Table 3 shows a comparison between the proposed scheme and other similar schemes from the security point of view. Protocols that are resistant to attacks and provide malicious revocation users from the system are more secure protocols.

5.4 Performance evaluation

In this section, the proposed protocol is compared with the previous similar schemes from the point of view of performance. Performance evaluation is performed based on the experimental results of the Study [36]; In this study for that purpose, was

```
(****************************** KGC ******************************)
let KGC =
let Kpub = EPM(Kpr, P) in
let tpub = EPM(tpr, P) in
let Beta = BL0(P, P) in
out (CH_P, (Kpub ,tpub ,Beta));
out (CH_S, (tpr));
in (CH_S, (IDA: bitstring));
new w: bitstring;
let Psda = EPM(w, P) in
let Psda' = XOR(IDA, (H1(MUL(w, Kpub)))) in
let PsdA = CON(Psda, Psda') in
let QPsdA = H2(PsdA) in
let MA = MUL(Kpr, QPsdA) in
out (CH_S, (PsdA, MA));
in (CH_S, (IDB: bitstring));
let Psdb = EPM(w, P) in
let Psdb' = XOR(IDB, (H1(MUL(w, Kpub)))) in
let PsdB = CON(Psdb, Psdb') in
let QPsdB = H2(PsdB) in
let MB = MUL(Kpr, QPsdB) in
out (CH_S, (PsdB, MB));
0.
```

(b) The KGC process part of the ProVerif code.

Fig. 3 (continued)

used the MIRACL library [39] that the system specifications for implementation are shown in Table 4.

Here, performance evaluation is done precisely as same as the study [27]. Also, for convenience, the symbols in Table 5 are used to perform performance comparisons. The relative times for cryptographic operations is given in Fig. 5. Table 6 summarizes the number of cryptographic operations related to signature verification in the proposed scheme and other similar schemes. Figure 6, shows the total running time of the proposed system and other similar protocols in two modes of the original validation algorithm and server-aided validation algorithm.

According to Fig. 5, the executing time of the bilinear pairing (5.275 ms) and scalar multiplication (1.970 ms) is much higher than the other operations. For this reason, with the fewer number of bilinear pairings and scalar multiplication operations, the efficiency of the protocol increases.

According to Table 6, the proposed scheme compared to the protocol [24] reduces the number of bilinear pairings in the signature verification sever-aided (from 1 to 0). Also, the proposed scheme compared to the protocol [27] reduces the number of scalar multiplication operations in the signature verification sever-aided (from 4 to 3). Therefore as shown in Fig. 6, the proposed scheme leads to an increase in the speed (efficiency) for signature validation server-aided to 6.918 ms.

```
(******************************** Server ********************************)
let Server =
in (CH_P, (PsdA: bitstring, T1: bitstring));
in (CH_S, (tpr: bitstring));
in (CH_P, (ML: bitstring));
if (ML <> PsdA) then
let QupdA = H3(PsdA, T1) in
let TupdA = MUL(tpr, QupdA) in
out (CH_P, (TupdA));
in (CH_P, (PsdB: bitstring, T2: bitstring));
if (ML <> PsdB) then
(
let QupdB = H3(PsdB, T2) in
let TupdB = MUL(tpr, QupdB) in
out (CH_P, (TupdB))
)
else 0.
(******************************** Alice ********************************)
let Alice =
in (CH_P, (Kpub: bitstring, tpub: bitstring, Beta: bitstring));
out (CH_S, (IDA));
in (CH_S, (PsdA: bitstring, MA: bitstring));
new T1: bitstring;
out (CH_P, (PsdA, T1));
in (CH_P, (TupdA: bitstring));
event begin_Alice(IDA);
in (CH_P, (PsdB: bitstring, TID: bitstring, AmountT: bitstring, Beta: bitstring));
let alphaA = EXP(Beta, FA) in
let vA = H4(PsdB, TID, AmountT, alphaA) in
let UA = SUM((EPM(FA, P)), (MUL(vA, (SUM(MA, TupdA))))) in
let sigmaA = CON(UA, alphaA) in
out (CH_P, (sigmaA, T1, PsdA, TID, PsdB, AmountT));
in (CH_P, (sigmaB: bitstring, T2: bitstring, PsdB: bitstring, TID: bitstring, AmountR:
bitstring, alphaB: bitstring, UB: bitstring, QupdB: bitstring, QPsdB: bitstring));
let v1 = H4(PsdB, TID, AmountR, alphaB) in
let e1 = (BL0(UB, P)) in
let r1 = MUL1(alphaB, BL(QPsdB, Kpub, v1), BL(QupdB, tpub, v1)) in
let Kij = r1 in
if (e1 = r1) then
event end_Alice(IDA)
else 0.
(******************************** Bob ********************************)
let Bob =
in (CH_P, (Kpub: bitstring, tpub: bitstring, Beta: bitstring));
out (CH_S, (IDB));
in (CH_S, (PsdB: bitstring, MB: bitstring));
new T2: bitstring;
out (CH_P, (PsdB, T2));
in (CH_P, (TupdB: bitstring));
event begin_Bob(IDB);
out (CH_P, (PsdB, TID, AmountT));
in (CH_P, (sigmaA: bitstring, T1: bitstring, PsdA: bitstring, UA: bitstring, alphaA:
bitstring, QupdA: bitstring, QPsdA: bitstring, QupdB: bitstring, QPsdB: bitstring));
let v2 = H4(PsdB, TID, AmountT, alphaA) in
let e2 = (BL0(UA,P)) in
let r2 = MUL1(alphaA, BL(QPsdA, Kpub, v2), BL(QupdA, tpub, v2)) in
let Kij = r2 in
if (e2 = r2) then
let Receipt = CON2(TID, AmountR, PsdB) in
let alphaB = EXP(Beta, FB) in
let vB = H4(PsdB, TID, AmountR, alphaB) in
let UB = SUM((EPM(FB, P)), (MUL(vB, (SUM(MB, TupdB))))) in
let sigmaB = CON(UB, alphaB) in
out (CH_P, (sigmaB, T2, PsdB, TID, AmountR));
event end_Bob(IDB)
else 0.
process     ((! KGC) | (! Server) | (! Alice) | (! Bob))
```

(c) The code snippet related to process part of validation algorithm original.

Fig. 3 (continued)

```
(********************************* Server *********************************)
let Server =
in (CH_P, (PsdA: bitstring,T1:bitstring));
in (CH_S, (tpr: bitstring));
in (CH_P, (ML: bitstring));
if (ML <> PsdA) then
let QupdA = H3(PsdA, T1) in
let TupdA = MUL(tpr, QupdA) in
out (CH_P, (TupdA));
in (CH_P, (sigmaB': bitstring, T2: bitstring, PsdB: bitstring, tpub: bitstring, QupdB:
bitstring, Kpub: bitstring, QPsdB: bitstring));
let Beta3 = (BL0(sigmaB', P)) in
let Beta4 = MUL(BL0(QPsdB, Kpub), BL0(QupdB, tpub)) in
out (CH_P, (Beta3, Beta4));
in (CH_P, (PsdB: bitstring, T2: bitstring));
if (ML <> PsdB) then
(
let QupdB = H3(PsdB, T2) in
let TupdB = MUL(tpr, QupdB) in
out (CH_P, (TupdB));
in (CH_P, (sigmaA': bitstring, T1: bitstring, PsdA: bitstring, QupdA: bitstring, QPsdA:
bitstring));
let Beta1 = (BL0(sigmaA', P)) in
let Beta2 = MUL(BL0(QPsdA, Kpub), BL0(QupdA, tpub)) in
out (CH_P, (Beta1, Beta2))
)
else 0.
(********************************* Alice *********************************)
let Alice =
in (CH_P, (Kpub: bitstring, tpub: bitstring, Beta: bitstring));
out (CH_S, (IDA));
in (CH_S, (PsdA: bitstring, MA: bitstring));
new T1: bitstring;
out (CH_P, (PsdA, T1));
in (CH_P, (TupdA: bitstring));
event begin_Alice(IDA);
in (CH_P, (PsdB: bitstring, TID: bitstring, AmountT: bitstring, Beta: bitstring));
let alphaA = EXP(Beta, FA) in
let vA = H4(PsdB, TID, AmountT, alphaA) in
let UA = SUM((EPM(FA ,P)), (MUL(vA, (SUM(MA, TupdA))))) in
let sigmaA = CON(UA, alphaA) in
out (CH_P, (sigmaA, T1, PsdA, TID, PsdB, AmountT));
in (CH_P, (sigmaB: bitstring, T2: bitstring, PsdB: bitstring, TID: bitstring, AmountR:
bitstring, alphaB: bitstring, UB: bitstring, QupdB: bitstring, QPsdB: bitstring));
let v1 = H4(PsdB, TID, AmountR, alphaB) in
new x1: bitstring;
new y1: bitstring;
let sigmaB' = SUM((MUL(x1, UB)), (MUL(y1, P))) in
out (CH_P, (sigmaB', PsdB, T2));
in (CH_P, (Beta3: bitstring, Beta4: bitstring));
let r1 = MUL1((EXP(alphaB, x1)), (EXP(Beta4, (MUL(v1, x1)))), (EXP(Beta, y1))) in
let Kij = r1 in
if (Beta3 = r1) then
event end_Alice(IDA)
else 0.
(********************************* Bob *********************************)
let Bob =
in (CH_P, (Kpub: bitstring, tpub: bitstring, Beta: bitstring));
out (CH_S, (IDB));
in (CH_S, (PsdB: bitstring, MB: bitstring));
new T2: bitstring;
out (CH_P, (PsdB, T2));
in (CH_P, (TupdB: bitstring));
event begin_Bob(IDB);
out (CH_P, (PsdB, TID, AmountT));
in (CH_P, (sigmaA: bitstring, T1: bitstring, PsdA: bitstring, UA: bitstring, alphaA:
bitstring, QupdA: bitstring, QPsdA: bitstring, QupdB: bitstring, QPsdB: bitstring));
let v2 = H4(PsdB, TID, AmountT, alphaA) in
new x2: bitstring;
new y2: bitstring;
let sigmaA' = SUM((MUL(x2, UA)), (MUL(y2, P))) in
out (CH_P, (sigmaA', PsdA, T1));
in (CH_P, (Beta1: bitstring, Beta2: bitstring));
let r2 = MUL1((EXP(alphaA, x2)), (EXP(Beta2, (MUL(v2, x2)))), (EXP(Beta, y2))) in
let Kij = r2 in
if (Beta1 = r2) then
let Receipt = CON2(TID, AmountR, PsdB) in
let alphaB = EXP(Beta, FB) in
let vB = H4(PsdB, TID, AmountR, alphaB) in
let UB = SUM((EPM(FB, P)), (MUL(vB, (SUM(MB, TupdB))))) in
let sigmaB = CON(UB, alphaB) in
out (CH_P, (sigmaB, T2, PsdB, TID, AmountR));
event end_Bob(IDB)
else 0.
process     ((! KGC) | (! Server) | (! Alice) | (! Bob))
```

(d) The code snippet related to process part of validation algorithm server-aided.

Fig. 3 (continued)

```
(******************************** Events ********************************)
event begin_Alice(bitstring).
event end_Alice(bitstring).
event begin_Bob(bitstring).
event end_Bob(bitstring).
(******************************** Queries ********************************)
free Kij: bitstring [private].
query attacker(Kij).
query id : bitstring ; inj-event(end_Alice(id)) ==>
        inj-event(begin_Alice(id)).
query id : bitstring ; inj-event(end_Bob(id)) ==>
        inj-event(begin_Bob(id)).
```

(e) The main part of the ProVerif code.

Fig. 3 (continued)

Table 2 Symbols of the proposed protocol

Symbol	Definition
P_{params}	Public parameters
K_{pr}	KGC's private key
K_{pub}	KGC's public key
t_{pr}	Time's private key
t_{pub}	Time's public key
ID_A	Alice's real identity
ID_B	Bob's real identity
$P\text{sd}_A$	Alice's pseudo-identity
Psd_B	Bob's pseudo-identity
M_A	Alice's partial private key
M_B	Bob's partial private key
F_A	Alice's full private key
F_B	Bob's full private key
T_{ID}	Transaction identity
T_i	ith time period
T_{upd_A}	Alice's time key
T_{upd_B}	Bob's time key
Amount_T	Amounts transferred
Amount_R	Amounts received
H_1, H_2	A map-to-point hash function
h	An ordinary hash function
σ	Digital signature
$\oplus$	An XOR operation

```
RESULT inj-event(end_Bob(id)) ==> inj-event(begin_Bob(id)) is true.
RESULT inj-event(end_Alice(id)) ==> inj-event(begin_Alice(id)) is true.
RESULT not attacker(Kij[]) is true.
```

Fig. 4 Results from the implementations of the proposed scheme using ProVerif

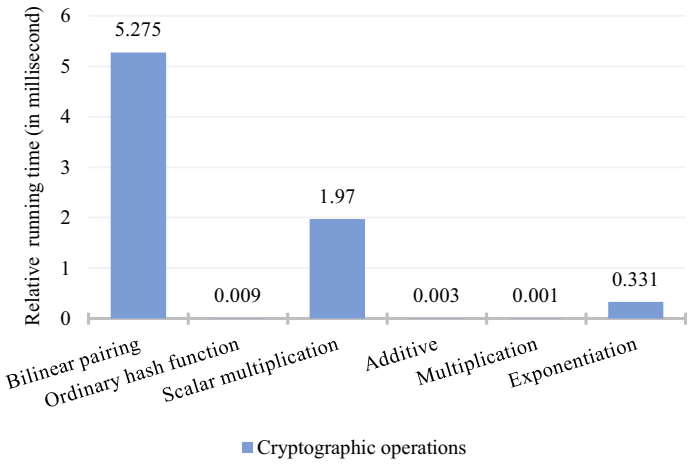

Fig. 5 Compare of relative times of cryptographic operations

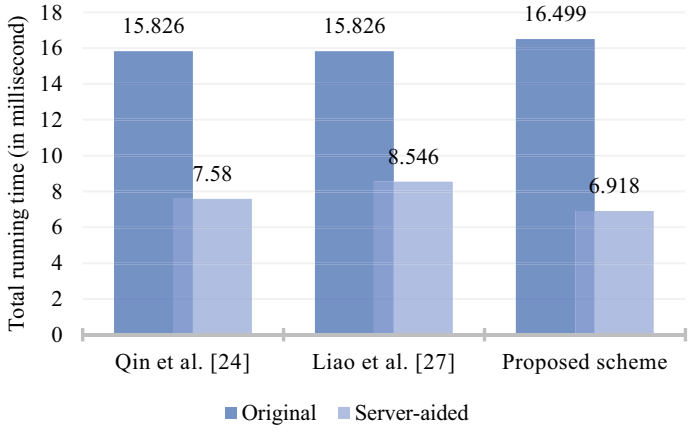

Fig. 6 Performance comparisons based on signature validation algorithms

Table 3 Security comparison

Protocols	Security properties						
	Unforgeability	Anonymity	Traceability	Revocable key	Non-repudiation	Secure against colluding attack	Secure against known attacks
Qin et al. [24]	✓	✓	✓	✗	✓	✗	–
Liao et al. [27]	✓	✓	✓	✗	✓	✓	–
Proposed scheme	✓	✓	✓	✓	✓	✓	✓

Table 4 System specifications for implementation

Platform	Specifications
Elastic Compute Service (ECS) host	Alibaba Cloud
The operating system of the host	Ubuntu 14.04 for 64 bit
CPU	E5-2630 0 @ 2.30 GHz
RAM	1 GB

Table 5 Symbols for performance comparisons

Symbol	Definition
N_{BL}	Number of executing a bilinear pairing
N_{OH}	Number of executing an ordinary hash function
N_{SMUL}	Number of executing a scalar multiplication operation
N_{SUM}	Number of executing an additive operation
N_{MUL}	Number of executing a multiplication operation
N_{EXP}	Number of executing an exponentiation operation

Table 6 The number of cryptographic operations for signature verification

Protocols		N_{BL}	N_{OH}	N_{SMUL}	N_{SUM}	N_{MUL}	N_{EXP}
Qin et al. [24]	Original	3	0	0	0	1	0
	Server-aided	1	0	1	1	1	1
Liao et al. [27]	Original	3	0	0	0	1	0
	Server-aided	0	0	4	1	1	2
Proposed scheme	Original	3	1	0	0	3	2
	Server-aided	0	1	3	1	3	3

6 Conclusion

The nature of the open media in wireless communications and the limited resources of mobile devices has led to great security challenges in mobile payment systems. In this paper, we proposed a new mobile payment system through an identity-based unforgeable signature. So that avoids additional costs due to the lack of digital certificates and provides non-repudiation and tracing capabilities. Also, this system provides an anonymity feature for users using the pseudo-identity method and provides key revocation uses the malicious user lists that contains the pseudo-identity of malicious users. To solve the problem of key escrow, we used the full private key of users (which was not known by the key generation center) for doing transactions in the scheme. In this plan, to reduce computational overhead, we outsourced some calculations to the cloud server, and also we used algorithms with less computational overhead. Thus, the proposed scheme

has increased the speed of the signature verification phase compared to previous schemes.

The security analysis showed that the proposed protocol provides more security properties than similar schemes. In this regard, we implemented the model using the ProVerif automatic tool, and the results showed that the proposed scheme is secure against active and passive attacks. Besides, the proposed design uses cloud-based logic technology to communicate between mobile devices and fast deployment of the protocol.

Acknowledgements The authors sincerely thank this journal for giving chances to proposing the scheme.

References

1. Boden J, Maier E, Wilken R (2020) The effect of credit card versus mobile payment on convenience and consumers' willingness to pay. JRCS 52:101910. https://doi.org/10.1016/j.jretconser.2019.101910
2. Isaac JT, Zeadally S (2014) Secure mobile payment systems. IT Prof 16:36–43. https://doi.org/10.1109/MITP.2014.40
3. Bhardwaj A, Subrahmanyam GVB, Avasthi V, Sastry H (2016) Security Algorithms for cloud computing. Procedia Comput Sci 85:535–542. https://doi.org/10.1016/j.procs.2016.05.215
4. Verma TBAK (2017) Data security in mobile cloud computing paradigm: a survey, taxonomy and open research issues. J Supercomput 73:2558–2631. https://doi.org/10.1007/s11227-016-1945-y
5. Tso R, Yi X, Huang X (2011) Efficient and short certificateless signatures secure against realistic adversaries. J Supercomput 55:173–191. https://doi.org/10.1007/s11227-010-0427-x
6. Shamir A (1985) Identity-based cryptosystems and signature schemes LNCS 84:47–53. https://doi.org/10.1007/3-540-39568-7_5
7. Dev D, Baishnab KL (2014) A review and research towards mobile cloud computing. API. https://doi.org/10.1109/MobileCloud.2014.41
8. Dahlberg T, Guo J, Ondrus J (2015) A critical review of mobile payment research. Electron Commer Res Appl 14:265–284. https://doi.org/10.1016/j.elerap.2015.07.006
9. Chaum D (1983) Blind signatures for untraceable payments. Adv Crypto 199:199–203. https://doi.org/10.1007/978-1-4757-0602-4_18
10. Chang C, Lai Y (2003) A flexible date-attachment scheme on e-cash. Comput Secur 22:160–166. https://doi.org/10.1016/S0167-4048(03)00214-1
11. Juang WS (2007) D-cash: a flexible pre-paid e-cash scheme for date-attachment. Electron Commer Res Appl 6:74–80. https://doi.org/10.1016/j.elerap.2005.12.001
12. Fan C, Guan DJ, Wang C, Lin D (2009) Cryptanalysis of Lee-Hwang-Yang blind signature scheme. Comput Stand Interfaces 31:319–320. https://doi.org/10.1016/j.csi.2008.02.002
13. Desmedt Y, Odlyzko AM (1985) A chosen text attack on the RSA cryptosystem and some discrete logarithm schemes. LNCS 218:516–522. https://doi.org/10.1007/3-540-39799-X_40
14. Bisel LD (2007) The role of SSL in Cybersecurity. IT Prof 9:22–25. https://doi.org/10.1109/MITP.2007.41
15. Guan HJ (2009) The Research of SET-Based Electronic Payment System Model. 2009 Int Conf E-bus Inf Syst Secur EBISS 2009. https://doi.org/10.1109/EBISS.2009.5138128.
16. Frisby W, Moench B, Recht B, Ristenpart T (2012) Security Analysis of Smartphone Point-of-Sale Systems. Woot, pp 1–3. http://dl.acm.org/citation.cfm?id=2372399.2372403.
17. Leu FY, Huang YL, Wang SM (2015) A secure M-Commerce system based on credit card transaction. Electron Commer Res Appl 14:351–360. https://doi.org/10.1016/j.elerap.2015.05.001
18. Martínez-Peláez R, Toral-Cruz H, Ruiz J, Velarde-Alvarado P (2015) P2PM-pay: person to person mobile payment scheme controlled by expiration date. Wirel Pers Commun 85:289–304. https://doi.org/10.1007/s11277-015-2738-y

19. Hou M, Xu Q, Lin F (2012) An efficient certificate revocation and verification scheme from multi-Hashing. Compute 7:1437–1444. https://doi.org/10.4304/jcp.7.6.1437-1444
20. Hu Q, Asghar MR, Brownlee N (2019) Checking certificate revocation efficiently using certificate revocation guard. JISA 48:102356. https://doi.org/10.1016/j.jisa.2019.06.012
21. Isaac JT, Zeadally S (2012) An anonymous secure payment protocol in a payment gateway centric model. Procedia Comput Sci 10:758–765. https://doi.org/10.1016/j.procs.2012.06.097
22. Yang JH, Lin PY (2016) A mobile payment mechanism with anonymity for cloud computing. J Syst Softw 116:69–74. https://doi.org/10.1016/j.jss.2015.07.023
23. Paar C, Pelzl J (2010) Understanding cryptography A textbook for students and practitioners. Springer, Heidelberg, pp 1–239. https://doi.org/10.1007/978-3-642-04101-3.
24. Qin Zhen, Sun J, Wahaballa A, Zheng W, Xiong H, Qin Zhiguang (2017) A secure and privacy-preserving mobile wallet with outsourced verification in cloud computing. Comput Stand Interfaces 54:55–60. https://doi.org/10.1016/j.csi.2016.11.012
25. Huang X, Mu Y, Susilo W, Wong DS, Wu W (2012) Certificateless signatures: new schemes and security models. Comput J 55:457–474. https://doi.org/10.1093/comjnl/bxr097
26. Zhang C, Lu R, Lin X, Ho PH, Shen X (2008) An efficient identity-based batch verification scheme for vehicular sensor networks. API, pp 816–824. https://doi.org/10.1109/INFOCOM.2008.58.
27. Liao Y, He Y, Li F, Zhou S (2018) Analysis of a mobile payment protocol with outsourced verification in cloud server and the improvement. Comput Stand Interfaces 56:101–106. https://doi.org/10.1016/j.csi.2017.09.008
28. Boyen X (2008) A tapestry of identity-based encryption: practical frameworks compared. IJACT 1:3–21. https://doi.org/10.1504/IJACT.2008.017047
29. Penttinen JTJ (2017) Wireless communications security: solution for the Internet of Things. Wiley Online Library, pp 189–206. https://doi.org/10.1002/9781119084402.
30. Fu Y, Chen CS, Zhou H (2009) Smart phone for mobile commerce. Comput Stand Interfaces 31:740–747. https://doi.org/10.1016/j.csi.2008.09.016
31. Rodríguez-Hernández MC, Ilarri S (2015) Pull-based recommendations in mobile environments. Comput Stand Interfaces 44:185–204. https://doi.org/10.1016/j.csi.2015.08.002
32. Park S, Lee I (2019) Enhanced signature RTD transaction scheme based on Chebyshev polynomial for mobile payments service in IoT device environment. J Supercomput 75:4617–4637. https://doi.org/10.1007/s11227-018-2546-8
33. Badra M, Badra RB (2016) A lightweight security protocol for NFC-based mobile payments. Procedia Comput Sci 83:705–711. https://doi.org/10.1016/j.procs.2016.04.156
34. Ning J, Ming L, Yang H (2014) An anonymous e-rental protocol based on ID-based cryptography and NFC. J Supercomput 70:31–53. https://doi.org/10.1007/s11227-013-1051-3
35. Yang JH (2017) An electronic transaction mechanism using mobile devices for cloud computing. Wirel Pers Commun 94:713–724. https://doi.org/10.1007/s11277-016-3646-5
36. Jia X, He D, Zeadally S, Li LI (2017) Efficient revocable ID-based signature with cloud revocation server. API 5:2945–2954. https://doi.org/10.1109/ACCESS.2017.2676021
37. Blanchet B (2016) Modeling and verifying security protocols with the applied Pi Calculus and ProVerif. Foundations Trends Priv Secur 1:1–135. https://doi.org/10.1561/3300000004
38. Blanchet B, Smyth B, Cheval V, Sylvestre M (2017) Automatic cryptographic protocol verifier, user manual and tutorial. http://www.cs.bham.ac.uk/~bas/papers/ProVerif-manual-version-1.98pl1.pdf.
39. Scott M (2011) On the Efficient Implementation of Pairing-Based Protocols. LNCS 7089:296–308. https://doi.org/10.1007/978-3-642-25516-8_18