Article

LIGHTWEIGHT AND SECURE IoT-BASED PAYMENT PROTOCOLS FROM AN IDENTITY-BASED SIGNATURE SCHEME

Abubaker Wahaballa [1,2,*]

[1] Arab East Colleges; wahaballah@arabeast.edu.sa
[2] Sudan Technological University (STU);
[*] Correspondence: wahaballah@uestc.edu.cn;

Abstract: After the great success of Mobile wallet, the Internet of Things (IoT) leaves the door wide open for consumers to use their connected devices to access their bank accounts and perform routine banking activities from anywhere, anytime and with any device. However, consumers need to feel safe when interacting with IoT-based payment systems, and their personal information should be protected as much as possible. Unlike as usually done in the literature, in this paper, we introduce two lightweight and secure IoT-based payment protocols based on an identity-based signature scheme. We adopt a server-aided verification technique to construct the first scheme. This technique allows to outsource the heavy computation overhead on the sensor node to a cloud server while maintaining the user's privacy. The second scheme is built upon a pairing-free ECC-based security protocol to avoid the heavy computational complexity of bilinear pairing operations. The security reduction results of both schemes are held in the Random Oracle Model (ROM) under the discrete logarithm and computational Diffie-Hellman assumptions. Finally, we experimentally compare the proposed schemes against each other and against the original scheme on the most commonly used IoT devices: a smartphone, a smartwatch and the embedded device Raspberry Pi. Compared with existing schemes, our proposed schemes achieve significant efficiency in the term of communication and computational overheads

Keywords: IoT-based payment protocols; identity-based signature; server-aided verification; pairing-free security protocols

check for updates

Citation: Wahaballa, A. Lightweight and Secure IoT-based Payment Protocols from an Identity-Based Signature Scheme. *Preprints* **2022**, *1*, 0. https://doi.org/

1. Introduction

Financial services and the payment industry are constantly evolving to meet customer requirements and to create a competitive advantage by providing better banking and financial services, improving operational efficiency and reducing costs. After plastic cards have been successfully replaced by mobile wallet (m-Wallet) [1], the Internet of Things (IoT) leaves the door wide open for consumers to use their connected devices to access their bank accounts and perform routine banking activities from anywhere, anytime and with any device. For instance, a connected watch can be used by a customer to conduct payment at a store, while a driver can pay for parking and fuel via a connected car. Furthermore, more milk can be bought automatically through a connected refrigerator. All these payment scenarios are categorized in the so-called IoT-based payment systems. The trend toward IoT-based payment systems started in the last few years and accelerated in 2018 [2].

As one of the effective methods of IoT-based payment systems, an in-vehicle payment solution has recently been launched by two leading credit card companies: Visa and MasterCard. In this solution, the driver is alerted when he/she is near a smart parking meter or fuel pump. At payment time, the amount of the purchased service is displayed in the dashboard. Afterwards, the entire payment process is simply completed with just a touch of a button. In addition to in-vehicle payments, wearable payment systems [3,4] have innovatively integrated payment methods into wearable devices such as smartwatches, jewelry, wristbands, fitness bands and other adaptable wearables.

An IoT-based payment system offers substantial efficiency benefits for both buyers and sellers, and both individuals and businesses. The consumers get shorter transaction time with high comfort and fast real-time response. Furthermore, their financial habits will improve as the IoT-based payment system supports them to spend their money wisely. For example, to prevent oversupply, reordering only occurs when the product is running low. On the other hand, the traders can gain more customers by ensuring IoT-friendliness, automate their logistics processes using smart shelves and optimize checkout and customer service by accepting IoT wallets.

Along with the many benefits of the IoT-based payment system, the associated risks and threats [5,6] cannot be omitted. With so much payment data shared across many IoT devices and things, it is inevitable that hackers and malicious users will try to get access to these most valuable and vulnerable data.

1.1. Motivations

Most breaches have financial motivations. These breaches aim to obtain information such as financial account information, users' credentials , or online banking details. Once the users' credentials are stolen , hackers misuse them to gain illegal access the to victim's account, defraud institutions, or commit other financial crimes.

To ensure that clients derive the maximum benefits and enjoy the appealing features of IoT-based payment systems, credit card companies and their partners should consider the following challenges. *First*, it is essential to guarantee that no one is allowed to impersonate any customer or create a fake payment request to steal funds. Otherwise, forgery and fraud will abort the invention of IoT-based payments in its initial stages. *Second*, anonymity and privacy have a significant influence on a user's attitude toward IoT-based payment systems. To increase the usage, acceptance and adoption of IoT-based payment systems, the personal information of consumers should be protected as far as possible. Furthermore, consumers need to feel safe when interacting with IoT-based payment systems. *Third*, the IoT embedded devices are extremely limited in resources such as storage capacity, processing capabilities, communication bandwidth and battery lifetime. Therefore, traditional encryption and authentication methods cannot be directly applied on sensor nodes in the IoT-based payment scenarios. This issue could be resolved by outsourcing the heavy computation overhead on sensor nodes to cloud servers while maintaining the user's privacy. Another solution lies in adopting pairing-free ECC-based security protocols. In this paper, both solutions are adopted to introduce two lightweight and agile IoT-based payment protocols.

A digital signature [7] is a fundamental cryptographic primitive that offers non-repudiation, unforgeability and authenticity of electronic payment systems. However, the traditional public key-based digital signature suffers from the complex certificate management. Fortunately, due to its certificate-free property, the identity-based signature (IBS) [8,9] has been introduced to simplify the complexity and reduce the heavy cost of certificate management in traditional public key-based digital signatures. Therefore, it is interesting and challenging to design a lightweight and secure IoT-based payment method based on the idea of an identity-based signature.

1.2. Contribution

Motivated by the limited resources of IoT devices, we introduce two lightweight and secure IoT-based payment protocols based on an identity-based signature scheme, where at each run a different lightweight computation and communication costs to suit the varying capabilities of IoT devices with limited resources. Then, we experimentally compare the proposed schemes against each other and against existing schemes on the most commonly used IoT devices: a smartphone, a smartwatch and the embedded device Raspberry Pi. To summarize, the major contributions of this article are twofold

1. We introduce a lightweight and secure IoT-based payment scheme based on an identity-based signature scheme. To preserve the limited resources of IoT embedded

devices and to meet the IoT-based payment system's goal of conserving bandwidth consumption, energy and processing power, we adopt a server-aided verification technique to reduce the heavy verification overhead. We further provide a security analysis of our proposed protocol to examine its correctness and soundness.

2. We propose an alternative solution of using a server-aided verification technique in the IoT-based payment scheme by adopting a pairing-free ECC-based security protocol. We then experimentally compare our pairing-free IoT-based payment scheme against original and server-aided verification protocols. The security reduction results of the second proposed scheme are held in the Random Oracle Model (ROM) under the discrete logarithm assumption

The rest of this paper is organized as follows. In Section 2, we discuss works related to our study. A high-level description of the proposed protocols and mathematical backgrounds are given in Section 3. In Section 4, the details of the proposed IoT-based payment protocols are presented. Through Section 5 and Section 6, a security analysis and performance evaluation are analyzed, followed by the conclusion and future work in Section 7.

2. Related Work

Throughout time, cashless-based payment system has evolved several times from smart cards to smart phones and Internet banking. The current trends for cashless-based payment system includes the debit and credit cards, Samsung Pay, Google Pay, Apple Pay, Wechat Pay, AliPay, and many more mobile banking applications. This paper is primarily related to electronic payment systems, mobile wallets, micropayments, and contactless payment systems.

2.1. Electronic Payment Systems

The Financial Technology (FinTech) [10] is an industry that leverage new technologies to provide secure, instant and efficient financial services. Its services (e.g. mobile banking apps) have been widely adopted by financial institutions. However, as revealed by the recent study [11], the FinTech financial services are not as secure as we expected. The study discovered thousands of vulnerabilities in 693 banking apps across over 80 countries, many of which could cause serious consequences, such as sensitive information leakage (e.g., PIN code, user name or users' credentials). Once the users' credentials are stolen , hackers misuse them to gain illegal access the to victim's account, defraud institutions, and other such financial and identity crimes.

In academia, many electronic payment systems have been proposed [12–16]. These systems aim to deliver payments from consumers to merchants in the most effective, efficient and error-free way, particularly in combination with attractive security properties. Unfortunately, each system is limited in some aspect.

2.2. Contactless Payment

Google Wallet in 2011 launched The first (contactless) payment system. Subsequently, it was followed by both Apple and Samsung Pay in 2014 and 2015, respectively. Also in 2015, Android Pay was announced as a new contactless system. Traditional contactless payments use cards. Now, majority of them follow Europay, MasterCard, and Visa (EMV) contactless specifications [17]. In mobile contactless payment system, the user is allowed to use the virtual debit and credit card information to securely pay for the purchases in store with those cards by waving the smart phone in front of the Near Field Communication Point-Of-Sale (NFC-POS).

The contactless payment can be classified according to the amount of money transferred into two main types: micro and macro [18]. In Micro payment, the user makes a small contactless transaction with *touch-and-go* practice, while in the Macro payment , the user conducts a big amount transaction with *touch-and-confirm* practice.

Despite the great convenience brought by mobile contactless payment system, fraud remains a significant consumer concern. Recent research [19] showed that 38% of users have a strong feeling that contactless payments are insecure, and around half (51%) are very or extremely concerned about fraud. As a result, 30% of all users with contactless cards still don't use them.

2.3. Mobile Wallet

A mobile wallet (m-Wallet) is a virtual wallet that keeps payment card information on a mobile device. Mobile wallets are a convenient way for a consumer to conduct in-store, in-app or online payments. In 2017, Qin *et al.* [1] introduced a secure and privacy-preserving mobile wallet by incorporating the certificateless signature and pseudo identity technique. Their approach significantly reduces the computation overhead in a resource-limited mobile device by offloading the heavy computation overhead on the user side to the untrusted cloud server. However, their payment protocol is insecure against a collusion attack between the customer, Alice, and the cloud server at the server-aided verification phase as proved in [20]. Because the cloud server is untrusted, there is no way to verify whether the returned information is valid or not. By considering the benefits of certificate-free property, Yeh and Chen *et al.* [21,22] proposed IoT-based payment protocols based on certificateless cryptography primitives. However, despite the security proofs, Zirui *et al.* [23] proved that the Yeh [21] protocol is insecure against public key replacement attack, while Chen *et al.* [22] suffers from perfect forward secrecy and replay attacks as shown in [24]. The work by Şengel *et al.* [25] proposed a new mobile payment cryptographic solution model, but this solution stores client's/credentials in the memory of the device, however, keeping user's credentials (Private keys and PIN) in the memory of a handset is very risky as these credentials can be easily compromised. A fully offline transaction e-commerce system model based on mobile payment which includes the offline POS terminal, mobile device, payment center was proposed by Li *et al.* [26], but it has limitation in day-to-day customer–merchant transactions, as it requires additional POS terminal.

Despite the security requirements of IoT-based payment seem similar to those identified in the literature, the limited resources of IoT devices such as storage capacity, processing capabilities, communication bandwidth and battery lifetime make the problem very novel and challenging. Furthermore, to the best of our knowledge, none of the state-of-the-art systems are designed to operate over IoT devices with limited resources.

3. Preliminaries

In this section, we first describe the system model. Then, the basic definitions and assumptions that are needed in the remainder of the paper are presented.

3.1. System Model

The scope of this section is to provide an overview of our proposed model. Our system model consists of the following participants:

- *Buyer: Alice*. A consumer who wants to purchase products or services from the seller.
- *Seller: Bob*. A trader who offers goods or services for sale either online or in-store
- *Payment Service Provider* ($\mathcal{PSP}$): Responsible for ensuring payment security and privacy. The $\mathcal{PSP}$ performs cryptographic initialization and user management.
- *Cloud server provider* ($\mathcal{CSP}$): This entity performs server-aided verification to reduce the heavy computational overhead on the sensor node.

The interaction scenario between the above participants is expressed graphically in Figure 1. Alice and Bob communicate with each other and with the other entities using short-range wireless communication protocol standards, such as NFC (IEEE 802.11), Bluetooth (IEEE 802.15.1), Wi-Fi or Zig-Bee (IEEE802.15.4), whereas the $\mathcal{PSP}$ and $\mathcal{CSP}$ are located and operated in the fixed network. The players take part in the following three phases of an IoT-based payment system: the Initialization Phase, Payment Phase and Verification Phase. In the initialization phase, the $\mathcal{PSP}$ initializes and assigns the system

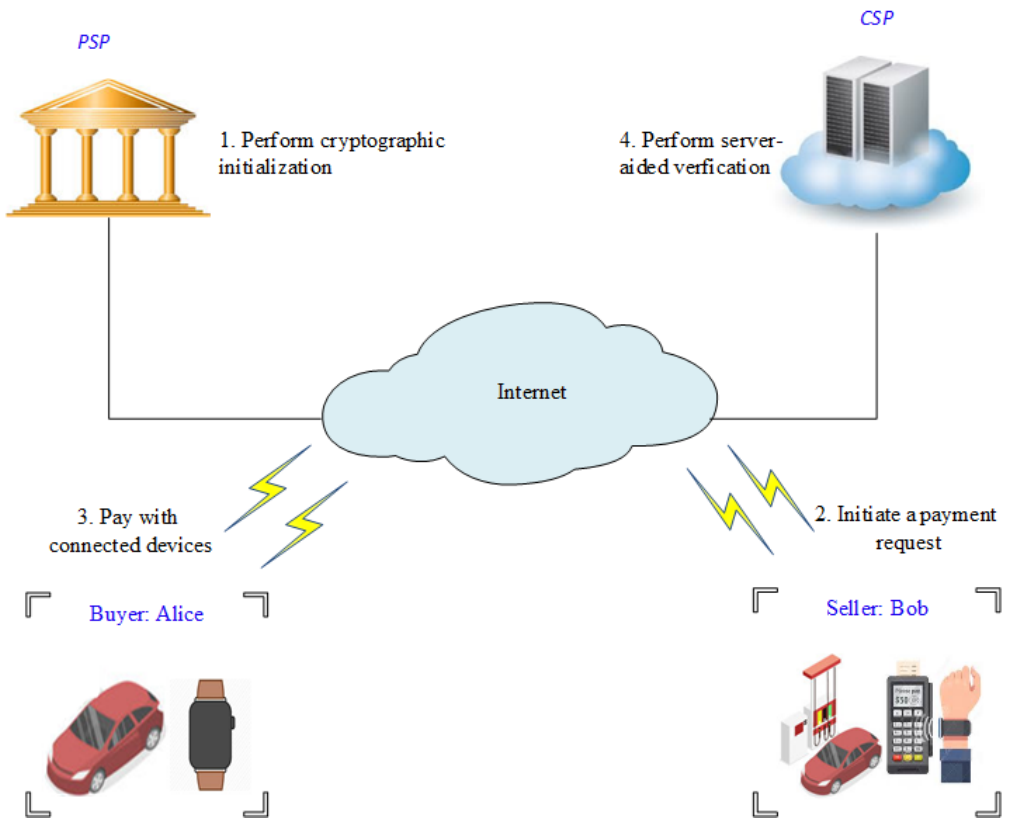
PSP
CSP
1. Perform cryptographic initialization
4. Perform server-aided verfication
Internet
3. Pay with connected devices
2. Initiate a payment request
Buyer: Alice
Seller: Bob

Figure 1. System Architecture.

parameters for each user. Then, it anonymously generates the user's pseudo identity using a tamper-proof device. In IoT-based payment systems, the IoT smart device alerts the user to purchase the necessary goods or services. For example, in our model, a smart refrigerator reminds Alice to buy more milk, whereas her smart vehicle notifies her when she is near a smart parking meter or fuel pump. In the payment phase, Alice uses her connected device to conduct the payment. There are three payment options that correspond to IoT smart devices: in-store, in-app or online. For an in-store payment, Alice tries to conduct a payment transaction using her wearable or hand-held device and Bob's Near Field Communication Point-Of-Sale (NFC-POS). To do so, Bob initiates a payment request on his NFC-POS. Afterward, Alice waves her wearable device near the NFC-POS, then she confirms the payment with her digital signature. For an in-app payment, Alice uses application program interfaces (APIs) to handle payments and transactions with just a touch of a button. For online payments, Alice preforms the payment transaction remotely over the Internet, using a wireless connection. Once Alice has completed the payment process, Bob uses Alice's public key to check the validity of Alice's signature and to ensure the payment integrity. Finally, after Alice's payment has passed the verification, Bob uses his private key to sign a receipt for the amount that he received.

3.2. Elliptic Curves Cryptography (ECC)

Miller and Koblitz [27,28] individually suggested the use of elliptic curves in cryptography in the middle of 1980s. Several years later, Elliptic Curves Cryptography (ECC) has attracted much attention from scientists, engineers and researchers worldwide. Today, ECC plays an important role in public key cryptography. Compared with RSA, ECC-based cryptosystems provide a high security level with a small key size and more efficient implementations [29]. Let $q > 3$ be a prime number and $E_q(a,b)$ be a non-singular elliptic curve over F_q, which can be defined as the Weierstrass form.

$$y^2 = x^3 + ax + b, \tag{1}$$

where $a, b, x, y \in F_q$ with the discriminant $\triangle = (4a^2 + 27b^2) \bmod q \neq 0$. A point $P(x,y)$ is an elliptic curve point if it satisfies Equation (1), and the point $Q(x,-y)$ is called the negative of P, i.e. $Q = -P$. The points $E_q(a,b)$ together with a point $\mathcal{O}$ (called point at infinity) form an additive cyclic group $\mathbb{G}$, that is, $\mathbb{G} = \{(x,y) : a,b,x,y \in F_q$ and $(x,y) \in E_q(a,b)\} \bigcup \{\mathcal{O}\}$ of prime order q. Scalar multiplication over $E|F_q$ can be computed as follows:

$$tP = P + P + ... + P_{(t \quad times)} \tag{2}$$

3.3. Bilinear Pairing

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic groups of the same large prime q. let g_1 be generators of $\mathbb{G}$. Assume that the discrete logarithm in $\mathbb{G}$ and $\mathbb{G}_T$ are hard. Let $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be an admissible pairing that satisfies the following properties:

1. *Bilinear*: Given $g_1, g_2, g_3 \in \mathbb{G}$, we have
   $\hat{e}(g_1, g_2 + g_3) = \hat{e}(g_1, g_2) \cdot \hat{e}(g_1, g_3)$
   For any $a, b, \in \mathbb{Z}_p^*$, we have
   $\hat{e}(ag_1, bg_2) = \hat{e}(g_1, g_2)^{ab} = \hat{e}(g_1, bg_2)^a = \hat{e}(ag_1, g_2)^b = \hat{e}(abg_1, g_2) = \hat{e}(g_1, abg_2)$;
2. *Non-degenerate*: There exists $g_1, g_2 \in \mathbb{G}$ such that $\hat{e}(g_1, g_2) \neq 1$. This means that if g_1 and g_2 are two generators of $\mathbb{G}$, then $\hat{e}(g_1, g_2)$ is a generator of $\mathbb{G}_T$;
3. *Computability*: There is an efficient algorithm to compute $\hat{e}(g_1, g_2)$ for all $g_1, g_2 \in \mathbb{G}$

3.4. Computational hardness assumption

Our scheme is based on the hardness assumptions as follows:

1. Discrete Logarithm (DL) Problem: Given a generator g_1 of a cyclic group $\mathbb{G}^*$ with order q, then compute $g_2 = g_1^a$ for a random $a \in \mathbb{Z}_q^*$.
2. Computational Diffie-Hellman (CDH) problem: Given (g, g^a, g^b) then compute g^{ab}, where $g \in \mathbb{G}$ is the generator and $a, b \in \mathbb{Z}_q^*$ is unknown.

3.5. Design goals

To maintain the IoT-based payment security, our protocols should be able to satisfy the following requirements:

1. **Unforgeability:** Only authorized users are allowed to make transactions. In other words, nobody can impersonate Alice or Bob to create a fake payment request or a fake or illegal receipt. Let $\mathcal{A}$ be a CDH attacker which attacks the IoT-based Payment Protocol and $\mathcal{F}$ be a forger who attacks the Identity-Based Signature Scheme (IBS). The forger $\mathcal{F}$ performs the IoT-based Payment Protocol instead of the $\mathcal{PSP}$ without to know the $\mathcal{PSP}$'s secret master-key. The attacker $\mathcal{A}$ plays the following game with $\mathcal{F}$.
   (a) ***Setup:*** *The $\mathcal{A}$ runs the setup algorithm to generate the system's parameters and sends them to the forger $\mathcal{F}$;*
   (b) ***Queries:*** *The forger $\mathcal{F}$ performs a series of queries to the following oracles:*
      - ***Extract query:*** *Key extraction oracle: returns private keys for arbitrary identities.*
      - ***Sign query:*** *Signature oracle: produces signatures on arbitrary messages using the private key corresponding to arbitrary identities.*
   (c) ***Forgery:*** *$\mathcal{F}$ produces a triple(ID^*, M^*,σ^*) made of an identity ID^*, whose private key was never extracted, and a message-signature pair (M^*,σ^*) such that(M^*,ID^*)*

was not submitted to the signature oracle. She wins if the verification algorithm accepts the triple(ID^*,M^*,σ^*).

2. **Anonymity:** The real identity of Alice only known to the $\mathcal{PSP}$ and Alice herself. Alice should be able to deal with Bob without disclosing her real identity. The same holds true when Bob signs a receipt.
3. **Traceability and Revocation:** The real identity of the malicious user must be traceable and revoked, but only by the $\mathcal{PSP}$. Malicious users must be prevented from accessing the tamper-proof device. Malicious users must be revoked and kicked-out of the payment protocol, but only by the $\mathcal{PSP}$.
4. **Unlinkability:** In our protocols, any user can conduct multiple payment transactions without others being able to link these transactions to a particular user. The proposed IoT-based payment protocols cannot be hacked by an attacker with any linkable information.
5. **Nonrepudiation:** Alice should not be able to deny the confirmed transaction. Bob also cannot deny that he received the amount paid by Alice.
6. **Resistance to Replay Attack:** An adversary cannot reuse the previously exchanged valid information to obtain the corresponding service.
7. **Resistance to Impersonation Attack:** An adversary cannot impersonate Alice or Bob to create a fake payment request or a fake or illegal receipt.
8. **Mutual Authentication:** Alice and Bob should authenticate each other before actual communication occurs for avoiding the potential malicious attacks.

Beside above security requirements, low energy and bandwidth consumption with other low-cost design capabilities are extremely desired properties in IoT-based payment systems.

4. Proposed Schemes

In this section, we concretely construct two lightweight IoT-based payment schemes. Both schemes are built upon an identity-based signature scheme. The first protocol is constructed based on Tzeng *et al.* scheme [30], and the second protocol relies on the Hu *et al.* scheme [31]. We employ server-aided verification protocols [32] to introduce the first scheme, while the second scheme is constructed based on a pairing-free approach. For convenience, the notations of the proposed scheme are defined in Table 1.

4.1. *Scheme I: IoT-based payment scheme with server-aided verification*

An IoT-based payment scheme with a server-aided verification consists of the following phases:

1. *Initialization Phase*
In this phase, the $\mathcal{PSP}$ initializes and assigns the system parameters for each user. These include two groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order q and a bilinear pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Let g_1 and g_2 denote two generators in $\mathbb{G}$. It next randomly selects its secret master-key $a \in_R \mathbb{Z}_q^*$ and computes its public master-key $P_{pub} = g_1^a$. It also selects two hash functions H_1 and H_2. For the first registration of Alice and Bob, $\mathcal{PSP}$ assigns real identities $ID_A, ID_B \in \mathbb{G}$ and passwords PW_A and PW_B for Alice and Bob respectively. $\mathcal{PSP}$ then stores $(a, ID_A, ID_B, PW_A, PW_B)$ into the tamper-proof device. Finally, $\mathcal{PSP}$ announces the public parameters: PP $= (\mathbb{G}, \mathbb{G}_T, q, \hat{e}, g_1, g_2, P_{pub}, \lambda_1, H_1, H_2)$, where $\lambda_1 = \hat{e}(g_1, g_1)$.
2. *Payment phase*
According to IoT device capabilities, there are three payment options: in-store, in-app or online. For in-store payments, Alice tries to conduct a payment transaction using her wearable or hand-held device and Bob's Near Field Communication Point-Of-Sale (NFC-POS), whereas APIs are used to handle payments and transactions with just a touch of a button in the in-app payment scenario. For online payments, Alice conducts a payment transaction remotely over the Internet, using a 4G/5G or WiFi wireless connection. The following steps illustrate how this can be done:

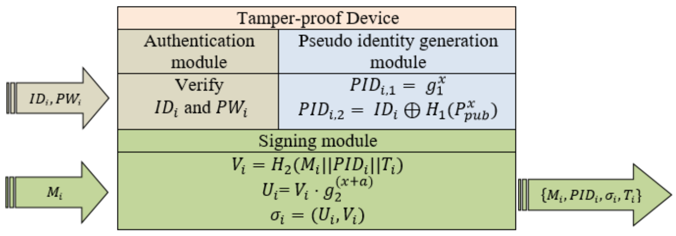
Tamper-proof Device
Authentication module
Pseudo identity generation module
Verify ID_i and PW_i
$PID_{i,1} = g_1^x$
$PID_{i,2} = ID_i \oplus H_1(P_{pub}^x)$
Signing module
$V_i = H_2(M_i||PID_i||T_i)$
$U_i = V_i \cdot g_2^{(x+a)}$
$\sigma_i = (U_i, V_i)$
ID_i, PW_i
M_i
$\{M_i, PID_i, \sigma_i, T_i\}$

Figure 2. The tamper-proof device for **Scheme I**.

Table 1. Notations

ACRONYM	DESCRIPTION
$\mathcal{PSP}$	Payment Service Provider
$\mathcal{CSP}$	Untrusted Cloud Server Provider
ID_A	Alice's real identity
PW_A	Alice's password
PID_A	Alice's anonymous identity
ID_B	Bob's real identity
PW_B	Bob's password
PID_B	Bob's anonymous identity
H_1, H_2	Two hash functions
σ	A signature on message M
PP	Public parameters
P_{pub}	The $\mathcal{PSP}$'s public master key
T	The current timestamp
$\hat{e}$	A bilinear pairing
$\oplus$	An Exclusive-OR (XOR)

- Step 1: Bob initiates a payment request on his NFC-POS, app or website by encoding the payment amount information into a message M.
- Step 2: Upon receiving the payment request, Alice logs in into the tamper-proof device using her real identity ID_A and password PW_A. If an incorrect ID_A or PW_A is entered, the tamper-proof device terminates the current process and refuses to perform further modules. On the other hand, if Alice passed the authentication module, then her real identity ID_A is transferred to the next module, the pseudo identity generation module. Figure 4 shows the procedures and modules of the tamper-proof device.
- Step 3: The pseudo identity generation module composes Alice's pseudo identity PID_A into $PID_{A,1}$ and $PID_{A,2}$. It then picks $x \in \mathbb{Z}_q^*$ and computes the pseudo identity PID_A as follows:

$$PID_{A,1} = g_1^x$$

$$PID_{A,2} = ID_A \oplus H_1(P_{pub}^x)$$

Finally, the pseudo identity generation module sets Alice's pseudo identity as: $PID_A = (PID_{A,1}, PID_{A,2})$.

- Step 4: Alice inputs the message M into the tamper-proof device. The tamper-proof device uses Alice's pseudo identity PID_A, x and a current timestamp T to generates the signature σ of M as follows:
  - Compute $V = H_2(M||PID_A||T)$
  - Compute $U = V \cdot g_2^{x+a}$

  Afterward, Alice obtains the final $\{M, PID_A, \sigma = (U, V), T\}$ and sends it to Bob.

3. *Verification Phase*
   Upon receiving the final message from Alice, Bob checks the validity of the signature to ensure the integrity of the payment information. To preserve the limited resources and help to meet the IoT-based payment system's goal of conserving bandwidth consumption, energy and processing power, we adopt a server-aided verification technique to minimize the number of pairing operations in the original verification process. The details of the original and server-aided verifications are described as follows.
   (a) *Original verification*: Given the final message $\{M, PID_A, \sigma = (U, V), T\}$ sent by Alice, Bob uses the public parameters $\mathrm{PP} = (\mathbb{G}, \mathbb{G}_T, q, \hat{e}, g_1, g_2, P_{pub}, \lambda_1, H_1, H_2)$ published by the $\mathcal{PSP}$ to check the validity of the signature as follows.
      - Step 1: To resist the replaying attack, Bob checks the freshness of the final message. Assume that the receiving time is T_B. Bob checks whether $\Delta T \geq T_B - T$. If it does, then Bob continues; otherwise, he rejects it.
      - Step 2: Bob verifies the signature by checking whether $\hat{e}(U, g_1) \stackrel{?}{=} \hat{e}(PID_{A,1} \cdot V \cdot P_{pub}, g_2)$ holds or not. If it does, output valid, otherwise output $\perp$.
   (b) *Server-aided verification*: As indicated in Step 2 during the original verification, two pairing operations are required to verify the signature. However, a bilinear pairing operation is computationally more expensive than other operations over elliptic curve groups. To minimize the number of pairing operations, Bob delegates an untrusted cloud server provider $\mathcal{CSP}$ to perform server-aided verification. The following steps and Figure 3 illustrate how Bob and $\mathcal{CSP}$ interact with each other.
      - Step 1: To check the freshness of the final message, Step 1 of the original verification is repeated here.
      - Step 2: Given public parameters $\mathrm{PP} = (\mathbb{G}, \mathbb{G}_T, q, \hat{e}, g_1, g_2, P_{pub}, \lambda_1, H_1, H_2)$ and the final message $\{M, PID_A, \sigma = (U, V), T\}$, Bob randomly chooses $r, t \in \mathbb{Z}_q^*$ and computes $U' = U \cdot g_1^r$ and $V' = V \cdot g_2^t$. He then sets $\sigma' = (U', V')$. Afterward, he sends the message and blind signature $\{PID_{A,1}, \sigma', T\}$ to the $\mathcal{CSP}$.
      - Step 3: Upon the $\mathcal{CSP}$ receiving $\{PID_{A,1}, \sigma', T\}$ from Bob, it computes $\lambda_2 = \hat{e}(U', g_1)$ and $\lambda_3 = \hat{e}(PID_{A,1} \cdot P_{pub}, V')$. It then sends λ_2, λ_3 to the Bob.
      - Step 4: Bob checks if $\lambda_2 \stackrel{?}{=} \lambda_3^{-t} \cdot \lambda_1^r$. If it does, output valid, otherwise output $\perp$.

4.2. ***Scheme II***: *Pairing-free IoT-based payment scheme*

In this section, our second pairing-free IoT-based payment scheme is introduced. This scheme is constructed based on Hu *et al.*'s scheme. Our second scheme consists of three phases as in **Scheme I**: the initialization phase, the payment phase and the verification phase.

1. *Initialization phase*
   Initially, the $\mathcal{PSP}$ inputs the security parameters k and determines the tuple $\{F_q, E|F_q, \mathbb{G}, P\}$ as defined in Section 3.2. Then, it picks secret master key $a \in \mathbb{Z}_q^*$ and computes its public master-key $P_{pub} = P^a$. Next, $\mathcal{PSP}$ chooses two hash

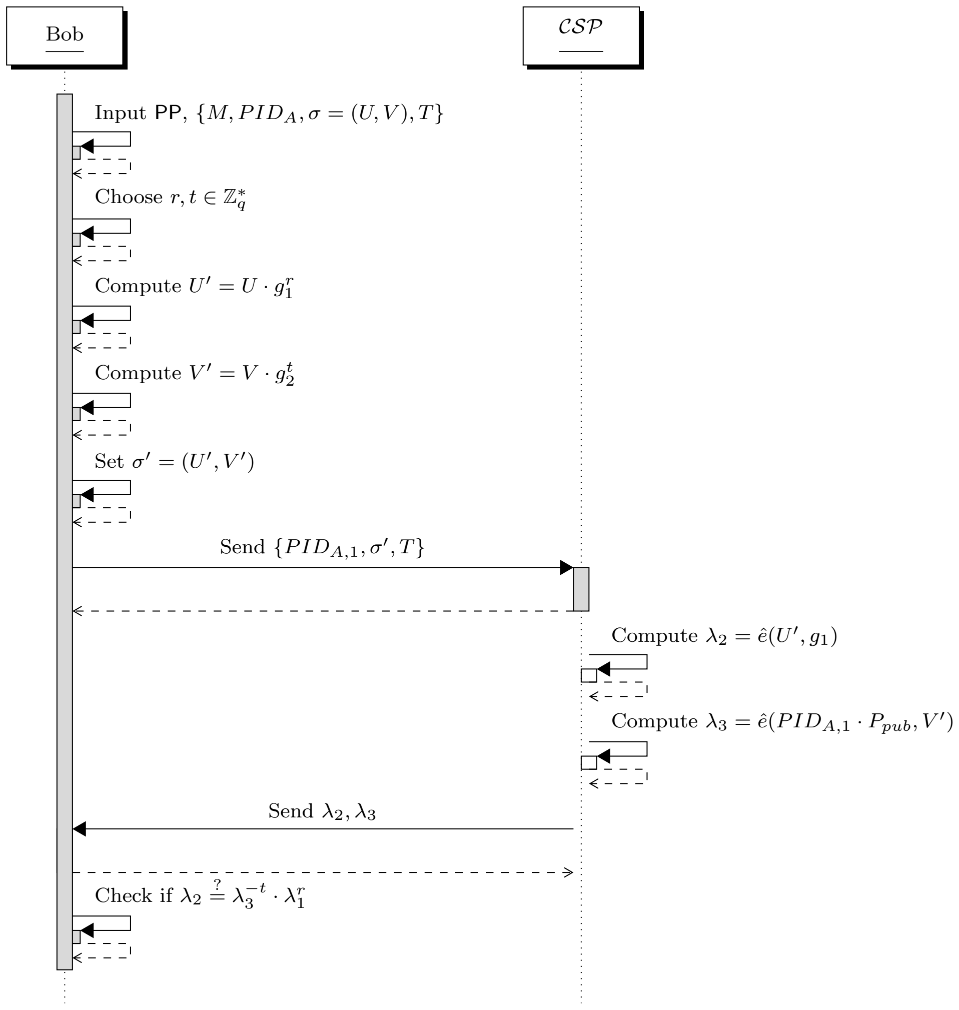
Bob
$\mathcal{CSP}$
Input PP, $\{M, PID_A, \sigma = (U, V), T\}$
Choose $r, t \in \mathbb{Z}_q^*$
Compute $U' = U \cdot g_1^r$
Compute $V' = V \cdot g_2^t$
Set $\sigma' = (U', V')$
Send $\{PID_{A,1}, \sigma', T\}$
Compute $\lambda_2 = \hat{e}(U', g_1)$
Compute $\lambda_3 = \hat{e}(PID_{A,1} \cdot P_{pub}, V')$
Send λ_2, λ_3
Check if $\lambda_2 \overset{?}{=} \lambda_3^{-t} \cdot \lambda_1^r$

Figure 3. Interaction between Bob and $\mathcal{CSP}$ in server-aided verification.

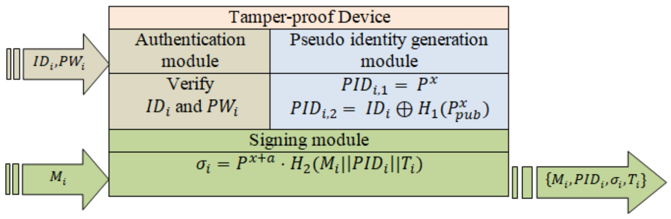
Tamper-proof Device
Authentication module
Pseudo identity generation module
Verify ID_i and PW_i
$PID_{i,1} = P^x$
$PID_{i,2} = ID_i \oplus H_1(P_{pub}^x)$
Signing module
$\sigma_i = P^{x+a} \cdot H_2(M_i||PID_i||T_i)$
ID_i, PW_i
M_i
$\{M_i, PID_i, \sigma_i, T_i\}$

Figure 4. The tamper-proof device for **Scheme II**.

functions H_1 and H_2. Finally, the $\mathcal{PSP}$ publishes the system parameters: PP $= (F_q, E|F_q, \mathbb{G}, P, P_{pub}, H_1, H_2)$.

2. *Payment Phase*
Steps 1 and 2 are the same as steps 1 and 2 of **Scheme I** during the same phase. The main differences are in step 3 and step 4.
   - Step 3: The pseudo identity generation module composes Alice's pseudo identity PID_A into $PID_{A,1}$ and $PID_{A,2}$. It then picks $x \in \mathbb{Z}_q^*$ and computes pseudo identity PID_A as follows:

$$PID_{A,1} = P^x$$

$$PID_{A,2} = ID_A \oplus H_1(P_{pub}^x)$$

Finally, the pseudo identity generation module sets Alice's pseudo identity as: $PID_A = (PID_{A,1}, PID_{A,2})$.

- Step 4: Alice inputs the message M into the tamper-proof device, The tamper-proof device uses Alice's pseudo identity PID_A, x and a current timestamp T to generate the signature σ of M as:

$$\sigma = P^{x+a} \cdot H_2(M||PID_A||T)$$

Next, Alice obtains the final message $\{M, PID_A, \sigma, T\}$ and sends it to Bob.

3. *Verification Phase*
Given the final message $\{M, PID_A, \sigma, T\}$ sent by Alice, Bob uses the public parameters PP $= (F_q, E|F_q, \mathbb{G}, P, P_{pub}, H_1, H_2)$ published by the $\mathcal{PSP}$ to check the validity of the signature as follows.
- Step 1: For freshness, Bob repeats step 1 of the original verification of **Scheme I**.
- Step 2: Bob verifies the signature by checking whether
$\sigma \overset{?}{=} PID_{A,1} \cdot H_2(M||PID_A||T)P_{pub}$ holds or not. If it does, output valid, otherwise output $\perp$.

5. Security Analysis

Considering the security requirements of an IoT-based payment system as a part of the system model of this article, in this section we discuss how these requirements can be achieved through both proposed protocols. The unforgeability property of each scheme is proved separately, while the other security properties are proved for both schemes together.

5.1. Correctness

For Scheme I, the correctness of signatures $\sigma = (U, V)$ is described as follows.

$$\begin{aligned}\hat{e}(U, g_1) =& \hat{e}(PID_{A,1} \cdot V \cdot P_{pub}, g_2)\\ =& \hat{e}(g_1^x \cdot V \cdot g_1^a, g_2)\\ =& \hat{e}(g_1^{x+a} \cdot V, g_2)\\ =& \hat{e}(g_2^{x+a} \cdot V, g_1)\\ =& \hat{e}(U, g_1)\end{aligned}$$

For Scheme II, the correctness of signatures σ is described as follows.

$$\begin{aligned}\sigma &= PID_{A,1} \cdot H_2(M||PID_A||T)P_{pub}\\ &= P^x \cdot H_2(M||PID_A||T)P_{pub}\\ &= P^x \cdot H_2(M||PID_A||T)P^a\\ &= P^{x+a} \cdot H_2(M||PID_A||T)\\ &= \sigma\end{aligned}$$

5.2. Security proof sketch

Theorem 1 (Unforgeability)**.** *Only authorized users are allowed to make transactions. In other words, nobody can impersonate Alice or Bob to create a fake payment request or a fake or illegal receipt. Let $\mathcal{A}$ be a CDH attacker which attacks the IoT-based Payment Protocol and $\mathcal{F}$ be a forger who attacks the Identity-Based Signature Scheme (IBS). The forger $\mathcal{F}$ performs the IoT-based Payment Protocol instead of the $\mathcal{PSP}$ without to know the $\mathcal{PSP}$'s secret master-key.*

Proof. Our **scheme I** is existentially unforgeable against adaptive chosen message attacks in the random oracle model if any adversary $\mathscr{A}$ with runtime t wins the game in detention with a probability at most ϵ after issuing at most q signing queries. Let $\mathscr{C}$ be a CDH attacker who receives a CDH challenge tuple (g_1, g_1^a, g_1^b) for $a, b \in \mathbb{Z}_q^*$ and $g_1 \in \mathbb{G}$. $\mathscr{A}$ interacts with $\mathscr{C}$ as simulated in the game that is defined in detention 1. We show how $\mathscr{C}$ may use $\mathscr{A}$ to solve the CDH problem.

Initial. Firstly, $\mathscr{C}$ calculates g_1^{ab} given g_1^a, g_1^b, for some unknown $a, b \in \mathbb{Z}_q^*$. $\mathscr{C}$ sets he public parameters including $P_{pub} = g_1^a$ and $g_2 = g_1^b$. Finally, $\mathscr{C}$ gives (g_1, g_2, P_{pub}) to the $\mathscr{A}$.

Queries. In this phase, $\mathscr{C}$ answers $\mathscr{A}$'s oracle queries as follows:

- H_2 queries. $\mathscr{C}$ randomly chooses $j \in [1, q_{H_2}]$, where q_{H_2} times H_2 queries. Whenever $\mathscr{A}$ requests the hash value of $(M_i||PID_{A_i}||T_i)$ for H_2, $\mathscr{C}$ does the following:
  1. $\mathscr{C}$ saves a list L^{list} which is initially empty.
  2. $\mathscr{C}$ tests whether $i = j$, if the value of $(M_i||PID_{A_i})$ already exists on the list L^{list} in a tuple $(M_i||PID_{A_i}||T_i||h_i)$, then $\mathscr{C}$ outputs $h_i = H_2(M_i||PID_{A_i}||T_i)$ as a response to $\mathscr{A}$'s query.
  3. Otherwise, if $i \neq j$, $\mathscr{C}$ picks $h_i \in \mathbb{Z}_q^*$ and sets $h_i = H_2(M_i||PID_{A_i}||T_i)$ and returns h_i to $\mathscr{A}$.
- Sign queries. Once $\mathscr{C}$ receives a signing query for message M_i from $\mathscr{A}$, $\mathscr{C}$ does the following:
  1. In spite of the fact that $\mathscr{C}$ does not know the private key, it can still produce the signature as follows. $\mathscr{C}$ looks for the tuple $(M_i||PID_{A_i}||T_i||h_i)$, if does not exist, $\mathscr{C}$ picks $\alpha_i, h_i \in \mathbb{Z}_q^*$ and $PID_{A_i,2} \in \mathbb{G}$. It then produces the signature as $U = g_2^{\alpha_i}$ and $PID_{A_i,1} = g_1^{\alpha_i}/h_i P_{pub}$. The validity of the produced signature $\{M_i, PID_{A_i}, \sigma_i = (U_i, h_i), T_i\}$ can be checked as follows.
     $\hat{e}(PID_{A_i,1} \cdot h_i P_{pub}, g_2) = \hat{e}(g_1^{\alpha_i}, g_2) = \hat{e}(U_i, g_1)$.
  2. If the tuple $(M_i||PID_{A_i}||T_i||h_i)$ already exists on the L^{list}, $\mathscr{C}$ picks another $\alpha_i, h_i \in \mathbb{Z}_q^*$ and $PID_{A_i,2} \in \mathbb{G}$, and produces the signature again. It then returns $\{M_i, PID_{A_i}, \sigma_i, T_i\}$ to the $\mathscr{A}$ and saves the tuple $(M_i||PID_{A_i}||T_i, h_i)$ in the L^{list}. For the adversary $\mathscr{A}$, all signatures produced by $\mathscr{C}$ are indistinguishable from those signatures computed by the legitimate user.

Forgery. Eventually, $\mathscr{C}$ receives two valid forged signature $\{PID_{A_i}, M_i^*, \sigma_i, T_i\}$ and $\{PID_{A_i}, M_i^*, \sigma_i^*, T_i^*\}$ with the same random "α-value" but different hash values in a polynomial time, where

$$\sigma_i = h_i \cdot g_2^{\alpha_i + a} \tag{3}$$

$$\sigma_i^* = h_i^* \cdot g_2^{\alpha_i + a} \tag{4}$$

From 3 and 4, $\mathscr{C}$ obtains the following equation:

$$(h_i - h_i^*)^{-1} \cdot (\sigma_i - \sigma_i^*) = g_2^a = g_1^{ab}$$

Finally, $\mathscr{C}$ has successfully obtained the solution of the CDH problem. However, this breaks the assumption that the CDH is hard. It implies that an attacker cannot impersonate any customer or forge the signature on a payment receipt. Therefore, the unforgeability, nonrepudiation and authenticity of payment receipt (message M) are achieved in our first proposed scheme. □

Proof. Our **scheme II** is existentially unforgeable against adaptive chosen message attacks in the random oracle model if and only if the DL is hard.

We describe how a challenger $\mathscr{C}$ can use an adversary $\mathscr{A}$ as a subroutine to solve a random given instance (g_1, g_2) to compute $g_2 = a g_1$ for a random $a \in \mathbb{Z}_q^*$ and $g_1, g_2 \in \mathbb{G}$.

Initial. $\mathscr{C}$ picks a random integer $s \in \mathbb{Z}_q^*$ and sets $P_{pub} = sg_1$. $\mathscr{C}$ gives (g_1, g_2, P_{pub}) to the $\mathscr{A}$.

Queries. In this phase, $\mathscr{A}$ makes a sequence of oracle queries. These queries are answered by $\mathscr{C}$ as in the unforgeability game in detention 1.

- H_1 queries. $\mathscr{C}$ randomly picks $j \in [1, q_{H_1}]$, where q_{H_1} times H_1 queries. Whenever $\mathscr{A}$ requests the hash value of $Q_i \in \mathbb{G}$ for H_1, $\mathscr{C}$ does the following:
    1. $\mathscr{C}$ keeps a list L^{list} which is initially empty.
    2. $\mathscr{C}$ tests whether $i = j$, if the value of $(Q_i||h_{1,i})$ already exists on the list L^{list} in a tuple $(M_i||PID_{A_i}||T_i||Q_i||h_{1,i}||h_{2,i})$, then $\mathscr{C}$ outputs $h_{1,i} = H_1(Q_i)$ as a response to the $\mathscr{A}$'s query.
    3. Otherwise, if $i \neq j$, $\mathscr{C}$ picks $h_{1,i} \in \mathbb{Z}_q^*$ and sets $h_{1,i} = H_1(Q_i)$ and returns $h_{1,i}$ to $\mathscr{A}$.
- H_2 queries. When $\mathscr{A}$ requests the hash value of $(M_i||PID_{A_i}||T_i)$ for H_2, $\mathscr{C}$ deals with this request as follows:
    1. If the value of $(M_i||PID_{A_i})$ already exists on the list L^{list} in a tuple $(M_i||PID_{A_i}||T_i||Q_i||h_{1,i}||h_{2,i})$, then $\mathscr{C}$ outputs $h_{2,i} = H_2(M_i||PID_{A_i}||T_i)$ as a response to the $\mathscr{A}$'s query.
    2. Otherwise, $\mathscr{C}$ picks $h_{2,i} \in \mathbb{Z}_q^*$ and sets $h_{2,i} = H_2(M_i||PID_{A_i}||T_i)$ and returns $h_{2,i}$ to $\mathscr{A}$.
- *Sign queries*. Upon $\mathscr{C}$ receiving a query on message M_i, it randomly picks $\alpha_i, \beta_i, \gamma_i \in \mathbb{Z}_q^*$ and computes the signature as follows.

$$\sigma_i = \alpha_i$$

$$PID_{A_i,2} = ID_{A_i} \oplus \beta_i$$

$$PID_{A_i,1} = g_1^{\alpha_i} / \gamma_i P_{pub}$$

The produced signature looks valid from $\mathscr{A}$'s view because

$$\begin{aligned} & g_1^{\sigma_i} \\ =& PID_{A_i,1} \cdot \gamma_i P_{pub} \\ =& g_1^{\alpha_i} / \gamma_i P_{pub} \cdot \gamma_i P_{pub} \\ =& g_1^{\alpha_i} = g_1^{\sigma_i} \end{aligned}$$

Forgery. According to the Forking Lemma [33], $\mathscr{C}$ can obtain two forged signatures $\{PID_{A_i}, M_i^*, \sigma_i, T_i\}$ and $\{PID_{A_i}, M_i^*, \sigma_i^*, T_i^*\}$. The forged signatures satisfy the following:

$$\sigma_i = h_i^{(\alpha_i + a)} \tag{5}$$

$$\sigma_i^* = h_i^{*(\alpha_i + a)} \tag{6}$$

From 5 and 6, $\mathscr{C}$ can obtain the solution of the given DL problem as follows.

$$(h_i - h_i^*)^{-1} \cdot (\sigma_i - \sigma_i^*) = a$$

□

Theorem 2 (Anonymity of scheme I and scheme II)**.** *In our proposed IoT-based payment schemes, scheme I and II, the real identity of Alice only known to the $\mathcal{PSP}$ and Alice herself. Alice* *should be able to deal with Bob without disclosing her real identity. The same holds true when Bob signs a receipt.*

Proof. As shown in pseudo identity generation module in Figure 4 and Step 3 during the payment phase, the real identity of Alice ID_A is converted into two anonymous and random pseudo identities $PID_A = (PID_{A,1}, PID_{A,2})$ for unknown $x \in \mathbb{Z}_q^*$. Consequently, the only way to determine the the real identity from the anonymous and pseudo identity pair is to know the secret master-key a from P_{pub} or the private key x from $PID_{A,1}$, as depicted in the traceability process below. Moreover, the anonymous identity pair $(PID_{A,1}, PID_{A,2})$ is an ElGamal-type ciphertext, which is secure against chosen-plaintext attacks (CPA). As a result, our proposed schemes provide the unconditional anonymity. □

Theorem 3 (Traceability)**.** *When a dispute occurs during the payment process, the PSP should be able to identify the disputing parties; it then tries to resolve the dispute, in case in which none of the disputing parties is a malicious party. Otherwise, the malicious parties are revoked and kicked-out of the payment protocol.*

Proof. Given the anonymous identity pair $(PID_{i,1}, PID_{i,2})$, only $\mathcal{PSP}$ can trace the real identity ID_i of the user as follows.

$$\begin{aligned} PID_{i,2} \oplus H_1(PID_{i,1}^a) =& ID_i \oplus H_1(P_{pub}^x) \oplus H_1(PID_{i,1}^a) \\ =& ID_i \oplus H_1(P^{x+a}) \oplus H_1(P^{a+x}) \\ =& ID_i \end{aligned}$$

It is clear that the $\mathcal{PSP}$ can easily trace the malicious users and find out their identities in order to revoke them from payment protocol. □

Theorem 4 (Revocation)**.** *Malicious users must be prevented from accessing the tamper-proof device. Malicious users must be revoked and kicked-out of payment protocol, but only by $\mathcal{PSP}$.*

Proof. If misbehavior is detected, the identity of the violating user can be easily traced by $\mathcal{PSP}$, as shown in the proof of Lemma 3. The $\mathcal{PSP}$ then deletes the malicious user information (ID_i, PW_i) from the tamper-proof device. Thus, the malicious users can be easily revoked and kicked-out of payment protocol in our proposed schemes. □

Theorem 5 (Unlinkability)**.** *For both schemes, any client can conduct multiple payment transactions, without others being able to link these transactions to a particular client.*

Proof. Each signature σ_i on message M_i partially consists of pseudo identity PID_i and timestamp T_i, which are changed every time. Furthermore, as shown in the proof of Lemma 2 and Lemma 3, it impossible to determine or even guess the real identity of user from his/her anonymous identity without prior knowledge of the secret master-key a and private key x. Therefore, there is no way to link two or more messages to a particular user, so the security property of unlinkability is achieved in both our schemes. □

Theorem 6 (Resistance to Impersonation Attack)**.** *An adversary cannot impersonate Alice or Bob to create a fake payment request or a fake or illegal receipt.*

Proof. Assume that an adversary attempts to send a valid payment request to Alice for impersonating as a legitimate merchant (Bob) to steal funds. To this end, it should generate the current timestamp and a signature for the message (payment request). However, to sign the message, the adversary needs to have the knowledge of master secret key $a \in_R \mathbb{Z}_q^*$, private key x and a current timestamp T. Any adversary cannot generate a valid payment request without these secrets. Similarly, no adversary can be successful to impersonate as the legitimate user (Alice). Thus, our protocols can resist the impersonation attack. □

Table 2. Cryptographic Operations Time in IoT devices

Operation	Time computation [μs]		
	Smartwatch	Smartphone	Pi 3
T_P	6571	1050	31
T_E	207	38	3.3

Smartwatch: HUAWEI Watch 2, Smartphone: HUAWEI P9 Lite 2017 and Pi 3: Raspberry Pi 3 Model B

Theorem 7 (Resistance to Replay Attack)**.** *The previously transmitted payment request and receipt cannot be reused by adversaries to obtain the corresponding responses from the legitimate entity.*

Proof. An adversary intends to prove its legitimate identity through delivering the previous valid signature of a legitimate entity to the receiver. However, in our protocol, this attack cannot be launched with success since the invalid timestamp condition results in the failure of authentication. Hence, our protocols can prevent the replay attack □

Theorem 8 (Mutual Authentication)**.** *In our protocols, mutual authentication dedicates the merchant (Bob) returns the confirmation message signed with its private key to the Buyer (Alice) if the received signature is valid, and vice versa.*

Proof. For **Scheme I**, given the final message $\{M, PID_A, \sigma = (U, V), T\}$ sent by Alice, Bob uses the public parameters PP $= (\mathbb{G}, \mathbb{G}_T, q, \hat{e}, g_1, g_2, P_{pub}, \lambda_1, H_1, H_2)$ published by the $\mathcal{PSP}$ to check the validity of the signature as follows.

- Step 1: To resist the replaying attack, Bob checks the freshness of the final message. Assume that the receiving time is T_B. Bob checks whether $\Delta T \geq T_B - T$. If it does, then Bob continues; otherwise, he rejects it.
- Step 2: Bob verifies the signature by checking whether $\hat{e}(U, g_1) \overset{?}{=} \hat{e}(PID_{A,1} \cdot V \cdot P_{pub}, g_2)$ holds or not. If it does, output valid, otherwise output $\perp$.

This verification can be performed as server-aided verification by cloud server provider $\mathcal{CSP}$ as shown in Figure 3.

For **Scheme II**, given the final message $\{M, PID_A, \sigma, T\}$ sent by Alice, Bob uses the public parameters PP $= (F_q, E|F_q, \mathbb{G}, P, P_{pub}, H_1, H_2)$ published by the $\mathcal{PSP}$ to check the validity of the signature as follows.

- Step 1: For freshness, Bob checks the freshness of the final message.
- Step 2: Bob verifies the signature by checking whether $\sigma \overset{?}{=} PID_{A,1} \cdot H_2(M||PID_A||T)P_{pub}$ holds or not. If it does, output valid, otherwise output $\perp$.

□

6. Performance Evaluation

6.1. Computation Overhead

To experimentally evaluate the performance of the proposed IoT-based payment protocols, we adopt the following most commonly used IoT devices: smartphone (HUAWEI P9 Lite 2017), smartwatch(HUAWEI Watch 2) and embedded device Raspberry Pi 3 Model B. The specifications of these devices are shown in Table 4. The performance evaluation is conducted based on the results in [34], in which the relative times of various cryptographic operations are averaged by repeated experiments in Java using the JPBC library [35].

For convenience, we adopt the following notations: T_p for bilinear pairing and T_E for exponentiation on $\mathbb{G}$ (implemented as scalar multiplication of an elliptic-curve point).

Table 3. Operations Time in IoT devices

Operation	Time computation [μs]		
	SmartWatch	Smartphone	Embedded Device
$T_{h(.)}$	176	85	479
T_{mul}	291	178	1743
T_{exp}	7521	5232	216269
T_e	1642230	1240235	3251354

Smartwatch: HUAWEI Watch 2, Smartphone: HUAWEI P9 Lite 2017 and Pi 3: Raspberry Pi 3 Model B

Table 4. IoT Devices Specifications

Device	Type	CPU	RAM	OS
HUAWEI Watch 2	Smartwatch	ARM Cortex-A7	768 MB	Android 7.0
HUAWEI P9 Lite 2017	Smartphone	Kirin 655	3 GB	Android 7.0
Raspberry Pi 3 Model B	IoT embedded board	ARM Cortex-A53	1 GB	Raspbian 9.3

Table 5. Efficiency Comparison in Smartwatch Platform

	Original Scheme*		Scheme I**		Scheme II†	
	Operation	Running time [μs]	Operation	Running time [μs]	Operation	Running time [μs]
Signing	$1T_E$	207	$1T_E$	207	$1T_E$	207
Total running time		207		207		207
Verifying	$2T_e$	3284460	$3T_{exp}$	22563	$1T_{h(.)}$	176
	$2T_{mul}$	582	$3T_{mul}$	873	$3T_{mul}$	873
Total running time in seconds		3285.052		23.436		1.049

*Original scheme that performs the original verification process.
** The proposed server-aided verification scheme.
† The proposed pairing-free ECC-based scheme.

Multiplication and hash function operations are omitted because of their small computational costs, compared to pairings and exponentiations. The consuming times of these operation are shown in Table 3. In the following, the efficiency of proposed server-aided and pairing-free schemes are compared against each other and against original scheme. From the efficiency comparison outlined in Table 5, 7 and 8, we figure out that in each device, the computational complexity for all the schemes are similar at the signing stage. However, the difference appears clearly during the verification process. As indicated in Table 5, the total running times in the Smartwatch platform of our schemes during the verification process are 23.436 seconds and 1.049 seconds for server-aided and pairing-free schemes respectively, whereas the original scheme needs 3285.052 at the same stage for the same device. From the results of the Smartphone platform that are detailed in Table 7, our server-aided and pairing-free schemes only require 16.23 and 0.619 seconds for the verification process, against 2480.826 seconds for the original scheme at same stage. The results in Table 8 also indicate that, compared with the original scheme, the total running times of our schemes are reduced by approximately 10 times and up to 1140 times for

Table 6. Efficiency Comparison in Smartphone Platform

	Original Scheme*		Scheme I**		Scheme II†	
	Operation	Running time [μs]	Operation	Running time [μs]	Operation	Running time [μs]
Signing	$1T_{h(.)}$	85	$1T_{h(.)}$	85	$1T_{h(.)}$	85
	$1T_{mul}$	178	$1T_{mul}$	178	$1T_{mul}$	178
	$1T_{exp}$	5232	$1T_{exp}$	5232	$1T_{exp}$	5232
Total running time in seconds		5.495		5.49		5.49
Verifying	$2T_e$	2480470	$3T_{exp}$	15696	$1T_{h(.)}$	85
	$2T_{mul}$	356	$3T_{mul}$	534	$3T_{mul}$	534
Total running time in seconds		2480.826		16.23		0.619

* Original scheme that performs the original verification process.
** The proposed server-aided verification scheme.
† The proposed pairing-free ECC-based scheme.

Table 7. Efficiency Comparison in Smartphone Platform

	Original Scheme*		Scheme I**		Scheme II†	
	Operation	Running time [μs]	Operation	Running time [μs]	Operation	Running time [μs]
Signing	$1T_{h(.)}$	85	$1T_{h(.)}$	85	$1T_{h(.)}$	85
	$1T_{mul}$	178	$1T_{mul}$	178	$1T_{mul}$	178
	$1T_{exp}$	5232	$1T_{exp}$	5232	$1T_{exp}$	5232
Total running time in seconds		5.495		5.49		5.49
Verifying	$2T_e$	2480470	$3T_{exp}$	15696	$1T_{h(.)}$	85
	$2T_{mul}$	356	$3T_{mul}$	534	$3T_{mul}$	534
Total running time in seconds		2480.826		16.23		0.619

* Original scheme that performs the original verification process.
** The proposed server-aided verification scheme.
† The proposed pairing-free ECC-based scheme.

server-aided and pairing-free schemes respectively, in embedded device Raspberry Pi 1 Model B for the verification process. Figures 5a, 5b and 5c, show the number of operations of our schemes versus the original scheme. It is worth mentioning that the original scheme is overshadowed by a heavy computational complexity of bilinear pairing operations.

6.2. Communication Overhead

The communication cost of our protocols and related payment protocols are analyzed in this subsection. We adopt the security level of 1024 bits RSA algorithm. In our experiment, we choose an Ate pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Since the size of in bilinear pairing operation is 512-bit (64 bytes), and the modulus size of q in ECC operation is 512 bits (Note that although the size of the elements in $\mathbb{G}$ is 160×16=2560 bits, the final size of one time scale multiplication on ECC should be 512 bits because of the modulus q). In addition, let the sizes of an identity, a timestamp, and a general hash function be 32 bits, 32 bits, and 160 bits. Then, a 8 characters 64 bits password is adopted in our protocols to defend against offline dictionary attacks. These parameters are summarized in the Table 9. The analysis process is shown below.

6.2.1. Scheme I (Server-aided verification Scheme)

1. **Registration phase:**

Table 8. Efficiency Comparison in Embedded Device Raspberry Pi 1 Model B.

	Original Scheme*		Scheme I**		Scheme II†	
	Operation	Running time [μs]	Operation	Running time [μs]	Operation	Running time [μs]
Signing	$1T_{h(.)}$	479	$1T_{h(.)}$	479	$1T_{h(.)}$	479
	$1T_{mul}$	1743	$1T_{mul}$	1743	$1T_{mul}$	1743
	$1T_{exp}$	216269	$1T_{exp}$	216269	$1T_{exp}$	216269
Total running time in seconds		218.491		218.491		218.491
Verifying	$2T_e$	6502708	$3T_{exp}$	648807	$1T_{h(.)}$	479
	$2T_{mul}$	3486	$3T_{mul}$	5229	$3T_{mul}$	5229
Total running time in seconds		6506.194		654.036		5.708

* Original scheme that performs the original verification process.
** The proposed server-aided verification scheme.
† The proposed pairing-free ECC-based scheme.

In this phase, $\mathcal{PSP}$ assigns real identities $ID_A, ID_B \in \mathbb{G}$ and passwords PW_A and PW_B for Alice and Bob respectively. The overhead of these messages are $2|\mathbb{G}| = 2 * 1024 + 2 * 64 = 2176$ bits = 272 bytes.

2. **Payment phase:**
In payment phase, Alice send the tuple $\{M, PID_A, \sigma = (U, V), T\}$ to Bob. From this tuple $PID_A, U, V \in \mathbb{G}$. Therefore, the communication cost for this phase is $3|\mathbb{G}| + 32 = 3 * 1024 + 32 + 256 = 3160$ bits = 395 bytes
3. **Server-aided verification**
In this phase, Bob sends the message and blind signature $\{PID_{A,1}, \sigma' = (U', V'), T\}$ to the $\mathcal{CSP}$, and he receives λ_2, λ_3 from $\mathcal{CSP}$. In these transmitted messages, $PID_{A,1}, U', V' \in \mathbb{G}$ and λ_2, λ_3 are $2|p|, |p|$. Hence, the communication overhead in this phase are $3|\mathbb{G}| + 2|p|, |p| = 3 * 1024 + 2 * 512 = 4,096$ bits= 512 bytes.

6.2.2. Scheme II (Pairing-free ECC-based Scheme)

1. **Registration phase:**
In this phase, $\mathcal{PSP}$ assigns real identities $ID_A, ID_B \in \mathbb{G}$ and passwords PW_A and PW_B for Alice and Bob respectively. The overhead of these messages are $2|\mathbb{G}| = 2 * 320 + 2 * 64 = 704$ bits = 88 bytes.
2. **Payment phase:**
In payment phase, Alice send the tuple $\{M, PID_A, \sigma, T\}$ to Bob. From this tuple $PID_A, \sigma \in \mathbb{G}$. Therefor, the communication cost for this phase is $2|\mathbb{G}| + 32 = 2 * 320 + 32 = 672$ bits = 84 bytes

To sum up, the overall communication overhead of our schemes are summarized in the Table 10. As shown in this Table, our proposed schemes have certain advantages on communication overhead. Specifically, our scheme II incurs lower communication overhead in comparison with the other two schemes [36,37]. Our scheme I is slightly more than Zeng *et al.*[36] scheme, but significantly less than Thammarat [37]. Furthermore, in our Scheme I, communication-computation trade-off is provided. As the processing capabilities of IoT devices is extremely limited, we adopt a server-aided verification technique to outsource the heavy computation overhead on the sensor node to a cloud server. This technique decreased the computation overhead by 89% in Raspberry Pi 3 Model B, and 99% in Smartphone and Smartwatch. But, on the other hand the communication overhead increased by 56%.

Table 9. Length of the group in bilinear pairing and ECC

System	Carve	Pairing	Cyclic group	$\|p\|, \|p\|$	$\|\mathbb{G}\|$	Length of Elements
Bilinear Pairing	$y^2 = x^3 + 1$	$\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$	$\mathbb{G}(P)$	$\|p\|$ 512 bits	q 160 bits	$|\mathbb{G}|$1024 bits
ECC	$y^2 = x^3 + ax$	None	$\mathbb{G}(P)$	$\|p\|$ 160 bits	q 160 bits	$|\mathbb{G}|$320 bits

Table 10. Comparison of communication costs (in bytes)

Scheme	Send a single message	Send n single messages
Zeng et al. [36]	172 + 132 + 328 + 20=625	652n
Thammarat [37]	424+152+448+243+484+243=1958	1958n
Our Scheme I	272+395+512= 1179	1179n
Our Scheme II	84+88=172	172n

6.3. Storage Overhead

In general, sensor node, IoT devices and smartcard have very less storage capacity than payment service provider ($\mathcal{PSP}$). Therefore, reduction of the requirement of the storage capacity of sensor node and IoT devices are essential for IoT-based applications. We kept this issue in designing our IoT-based payment protocols. To this end, in initialization phase, $\mathcal{PSP}$ stores $(a, ID_A, ID_B, PW_A, PW_B)$. These parameters are equivalent to $2|\mathbb{G}| + |\mathbb{Z}_q^*| + 2 * 64bit$, while sensor node stores nothing in its memory during all stages of protocol, but the user needs to memorize the password.

7. Conclusions

This paper has raised a challenging question: how to increase convenience and enable clients to derive maximum benefits and to enjoy the appealing features of IoT-based payment systems. To answer this question, we have proposed two lightweight and secure IoT-based payment protocols based on the identity-based signature scheme. The security analysis demonstrates that the proposed protocols are provably secure in the random oracle model under the discrete logarithm and computational Diffie-Hellman assumptions. In addition, the performance evaluation results show that the proposed protocols have lower computational and communication costs, which will enable us to satisfy the IoT-based payment system's goal of conserving bandwidth consumption, energy and processing power. The long-term results of this effort is to continue improve the communication and computation costs at the sensor node.

References

1. Qin, Z.; Sun, J.; Wahaballa, A.; Zheng, W.; Xiong, H.; Qin, Z. A Secure and Privacy-preserving Mobile Wallet with Outsourced Verification in Cloud Computing. *Comput. Stand. Interfaces* **2017**, *54*, 55–60.
2. Dione, D.; Riche, G.; Cho, J.; Hasson, R.; Rettaroli, P.; Valle, K.D. Enabling Commerce Across the Internet of Things, 2017.
3. Cha, B.R.; Lee, S.H.; Park, S.B.; Ji, Y.; et al. Design of micro-payment to strengthen security by 2 factor authentication with mobile & wearable devices. *Advanced Science and Technology Letters* **2015**, *109*, 28–32.
4. Zhou, T.T.; Zhou, D.T.; Zhou, A.H. One-touch payment using haptic control via a messaging and calling multimedia system on mobile device and wearable device, currency token interface, point of sale device, and electronic payment card, 2015. US Patent 8,985,442.
5. Bhardwaj, A.; Kaushik, K.; Kumar, M., Taxonomy of Security Attacks on Internet of Things. In *Security and Privacy in Cyberspace*; Kaiwartya, O.; Kaushik, K.; Gupta, S.K.; Mishra, A.; Kumar, M., Eds.; Springer Nature Singapore: Singapore, 2022; pp. 1–24. https://doi.org/10.1007/978-981-19-1960-2_1.
6. Wheelus, C.; Zhu, X. IoT network security: threats, risks, and a data-driven defense framework. *IoT* **2020**, *1*, 259–285.

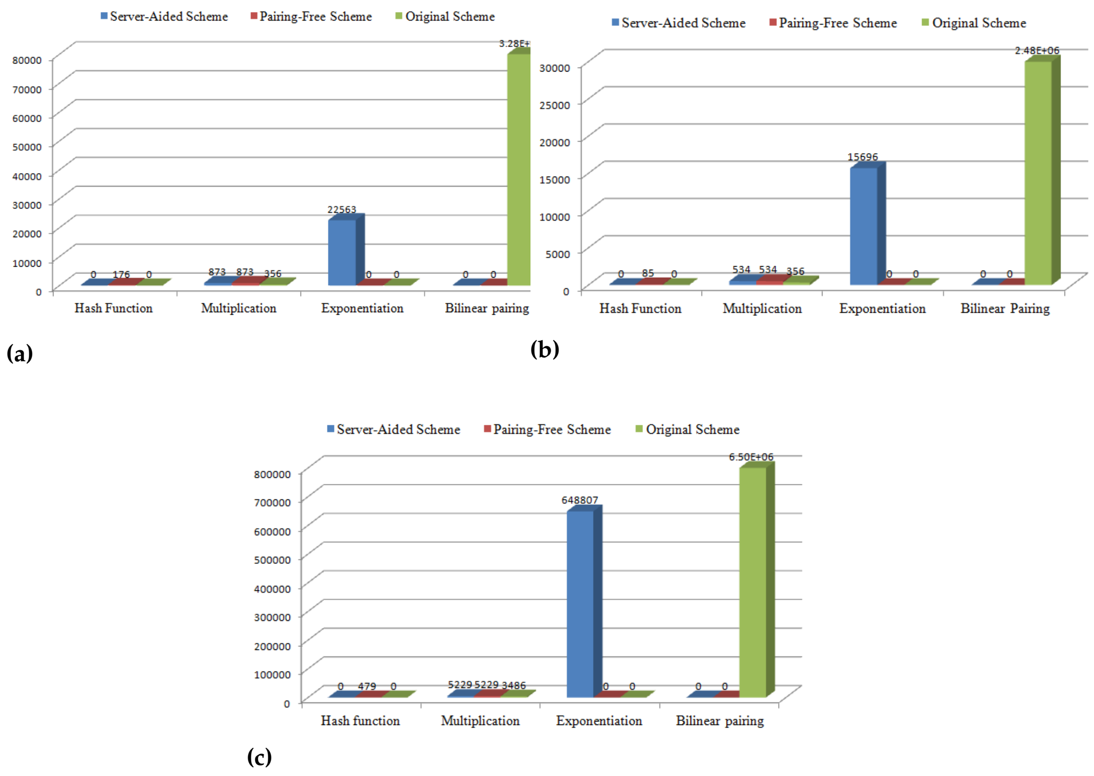

Figure 5. (a) Efficiency Comparison in Smartwatch Platform ; (b) Efficiency Comparison in Smartphone Platform; (c) Efficiency Comparison in Embedded Device Raspberry Pi 1 Model B.

7. Rivest, R.L.; Shamir, A.; Adleman, L. A Method for Obtaining Digital Signatures and Public-key Cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. https://doi.org/10.1145/359340.359342.
8. Paterson, K. ID-based signatures from pairings on elliptic curves. *Electronics Letters* **2002**, *38*, 1025–1026(1).
9. Hess, F. Efficient Identity Based Signature Schemes Based on Pairings. In Proceedings of the Revised Papers from the 9th Annual International Workshop on Selected Areas in Cryptography; Springer-Verlag: London, UK, UK, 2003; SAC '02, pp. 310–324.
10. Chen, C.C.; Liao, C.C. Research on the development of Fintech combined with AIoT. In Proceedings of the 2021 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), 2021, pp. 1–2. https://doi.org/10.1109/ICCE-TW52618.2021.9602952.
11. Chen, S.; Su, T.; Fan, L.; Meng, G.; Xue, M.; Liu, Y.; Xu, L. Are mobile banking apps secure? what can be improved? In Proceedings of the Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2018, pp. 797–802.
12. Hassan, M.A.; Shukur, Z.; Hasan, M.K. An efficient secure electronic payment system for e-commerce. *computers* **2020**, *9*, 66.
13. Wang, F.; Yang, N.; Shakeel, P.M.; Saravanan, V. Machine learning for mobile network payment security evaluation system. *Transactions on Emerging Telecommunications Technologies* **2021**, p. e4226.
14. Deng, X.; Gao, T. Electronic payment schemes based on blockchain in VANETs. *IEEE Access* **2020**, *8*, 38296–38303.
15. Tut, D. FinTech and the Covid-19 pandemic: Evidence from electronic payment systems. *Available at SSRN 3660987* **2020**.
16. Shanmugapriyan, J.; Parthasarathy, R.; Sathish, S.; Prasanth, S. Secure Electronic Transaction Using AADHAAR Based QR Code and Biometric Authentication. In Proceedings of the 2022 International Conference on Communication, Computing and Internet of Things (IC3IoT). IEEE, 2022, pp. 1–4.
17. Basin, D.; Sasse, R.; Toro-Pozo, J. The EMV Standard: Break, Fix, Verify. In Proceedings of the 2021 IEEE Symposium on Security and Privacy (SP), 2021, pp. 1766–1781. https://doi.org/10.1109/SP40001.2021.00037.
18. Gupta, B.B.; Narayan, S. A key-based mutual authentication framework for mobile contactless payment system using authentication server. *Journal of Organizational and End User Computing (JOEUC)* **2021**, *33*, 1–16.
19. Nilsson, H. Trust issues? The need to secure contactless biometric payment cards. *Biometric Technology Today* **2021**, *2021*, 5–8.
20. Liao, Y.; He, Y.; Li, F.; Zhou, S. Analysis of a mobile payment protocol with outsourced verification in cloud server and the improvement. *Computer Standards & Interfaces* **2018**, *56*, 101 – 106. https://doi.org/https://doi.org/10.1016/j.csi.2017.09.008.
21. Yeh, K.H. A secure transaction scheme with certificateless cryptographic primitives for IoT-based mobile payments. *IEEE Systems Journal* **2017**, *12*, 2027–2038.
22. Chen, Y.; Xu, W.; Peng, L.; Zhang, H. Light-weight and privacy-preserving authentication protocol for mobile payments in the context of IoT. *IEEE Access* **2019**, *7*, 15210–15221.

23. Qiao, Z.; Yang, Q.; Zhou, Y.; Zhang, M. Improved Secure Transaction Scheme With Certificateless Cryptographic Primitives for IoT-Based Mobile Payments. *IEEE Systems Journal* **2022**, *16*, 1842–1850. https://doi.org/10.1109/JSYST.2020.3046450.
24. Xiong, H.; Wu, Y.; Jin, C.; Kumari, S. Efficient and Privacy-Preserving Authentication Protocol for Heterogeneous Systems in IIoT. *IEEE Internet of Things Journal* **2020**, *7*, 11713–11724. https://doi.org/10.1109/JIOT.2020.2999510.
25. Şengel, Ö.; Aydin, M.A.; Sertbaş, A. A survey on white box cryptography model for mobile payment systems. In Proceedings of the International Telecommunications Conference. Springer, 2019, pp. 215–225.
26. Li, S.; Hu, X.; Fengling.; Zhang, Y.; Dong, W.; Ye, J.; Sun, H. Research on Offline Transaction Model in Mobile Payment System. In Proceedings of the Frontier Computing; Hung, J.C.; Yen, N.Y.; Hui, L., Eds.; Springer Singapore: Singapore, 2019; pp. 1815–1820.
27. Miller, V.S. Use of elliptic curves in cryptography. In Proceedings of the Advances in Cryptology CRYPTO 85 Proceedings. Springer, 1985, pp. 417–426.
28. Koblitz, N. Elliptic curve cryptosystems. *Mathematics of computation* **1987**, *48*, 203–209.
29. Bos, J.W.; Halderman, J.A.; Heninger, N.; Moore, J.; Naehrig, M.; Wustrow, E. Elliptic curve cryptography in practice. In Proceedings of the International Conference on Financial Cryptography and Data Security. Springer, 2014, pp. 157–175.
30. Tzeng, S.F.; Horng, S.J.; Li, T.; Wang, X.; Huang, P.H.; Khan, M.K. Enhancing Security and Privacy for Identity-Based Batch Verification Scheme in VANETs. *IEEE Transactions on Vehicular Technology* **2016**, *66*, 3235–3248.
31. Hu, X.; Wang, J.; Xu, H.; Liu, Y.; Zhang, X. Secure and Pairing-Free Identity-Based Batch Verification Scheme in Vehicle Ad-Hoc Networks. In Proceedings of the Intelligent Computing Methodologies: 12th International Conference, ICIC 2016, Lanzhou, China, August 2-5, 2016, Proceedings, Part III; Huang, D.S.; Han, K.; Hussain, A., Eds.; Springer International Publishing: Cham, 2016; pp. 11–20.
32. Xu, L.; Li, J.; Tang, S.; Baek, J. Server-Aided Verification Signature with Privacy for Mobile Computing. *Mobile Information Systems*, *2015*.
33. Pointcheval, D.; Stern, J. Security Arguments for Digital Signatures and Blind Signatures. *J. Cryptol.* **2000**, *13*, 361–396. https://doi.org/10.1007/s001450010003.
34. Malina, L.; Hajny, J.; Martinasek, Z. Privacy-preserving authentication systems using smart devices. In Proceedings of the 2016 39th International Conference on Telecommunications and Signal Processing (TSP), 2016, pp. 11–14. https://doi.org/10.1109/TSP.2016.7760820.
35. De Caro, A.; Iovino, V. jPBC: Java pairing based cryptography. In Proceedings of the Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011; , 2011; pp. 850–855.
36. Zeng, X.; Xu, G.; Zheng, X.; Xiang, Y.; Zhou, W. E-AUA: An efficient anonymous user authentication protocol for mobile IoT. *IEEE Internet of Things Journal* **2018**, *6*, 1506–1519.
37. Thammarat, C. Efficient and secure NFC authentication for mobile payment ensuring fair exchange protocol. *Symmetry* **2020**, *12*, 1649.