A trustworthy and untraceable centralised payment protocol for mobile payment

Article

Accepted Version

A Trustworthy and Untraceable Centralised Payment Protocol for Mobile Payment

JEYAMOHAN NEERA, Department of Computer and Information Sciences, Northumbria University, UK
XIAOMIN CHEN, Department of Computer Science, University of Reading, UK
NAUMAN ASLAM, Department of Computer and Information Sciences,Northumbria University, UK
BIJU ISSAC, Department of Computer and Information Sciences,Northumbria University, UK

Current mobile payment schemes gather detailed information about purchases customers make. This data can then be used to infer a customer's spending behaviour, potentially violating their privacy. To tackle this problem, we propose an untraceable mobile payment scheme that strikes a better balance, preserving user privacy while allowing the Third-Party Service Provider (TPSP) to collect necessary information such as card details and transaction amount for regulatory compliance. Our scheme offers untraceability for legitimate users from malicious adversaries and curious TPSPs using cryptographic primitives such as partially blind signatures, zero-knowledge proofs and identity-based signatures. It also guarantees that only authorised TPSPs can issue valid payment tokens, and even with limited data the TPSP can still prevent dishonest customers/merchants from double-spending a payment token. We also propose a comprehensive evaluation framework to assess the untraceable payment schemes against seven key criteria such as untraceability, exculpability - merchant double-spending, exculpability - customer double-spending, unforgeability, confidentiality, message authenticity, efficiency and regulatory compliance. We rigorously benchmark the security and privacy of our proposed payment scheme against this framework and other established schemes. Furthermore, we formally verify these properties using complexity-based analysis and Proverif modelling.

CCS Concepts: • **Security and privacy → Software and application security**; **Domain-specific security and privacy architectures**;

Additional Key Words and Phrases: Untraceability, Mobile Payment, Security, Privacy, Formal Analysis

ACM Reference Format:
Jeyamohan Neera, Xiaomin Chen, Nauman Aslam, and Biju Issac. 2018. A Trustworthy and Untraceable Centralised Payment Protocol for Mobile Payment. *J. ACM* 37, 4, Article 111 (August 2018), 29 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The proliferation of mobile devices has led to rapid growth in mobile commerce, enabling users to buy and sell goods and services using smartphones and other devices. Statistics indicate mobile commerce sales reached $3.56 trillion in 2022, demonstrating significant growth from previous years [40]. This growth is projected to continue, with estimates suggesting mobile commerce sales could exceed $7 trillion by 2025 [41]. Hence, mobile payment, a core element in mobile commerce,

Authors' addresses: Jeyamohan Neera, j.neera@northumbria.ac.uk, Department of Computer and Information Sciences, Northumbria University, UK; Xiaomin Chen, Department of Computer Science, University of Reading, UK, xiaomin.chen@reading.ac.uk; Nauman Aslam, Department of Computer and Information Sciences,Northumbria University, UK, nauman.aslam@northumbria.ac.uk; Biju Issac, Department of Computer and Information Sciences,Northumbria University, UK, biju.issac@northumbria.ac.uk.

ACM 0004-5411/2018/8-ART111
https://doi.org/XXXXXXX.XXXXXXX

has emerged as a significant research focus within industry and academia throughout the past decade.

Mobile payment systems allow users to initiate, authorise and finalise purchases. A variety of mobile payment options are available nowadays, including card-based payments, contactless payments and carrier billing [22]. These technologies often bypass the need for credit card centres or banks and rely on pre-registration with a Third-party Payment Service Provider (TPSP) which acts as an intermediary between customers and merchants. Popular examples of TPSPs include Google Pay, Apple Pay, Ali Pay, PayPal, and Venmo. TPSP-based mobile payment systems often store users' credit/debit card details, transaction-related information such as item details, merchant details and user login information. They build a comprehensive purchase profile of each customer using this information. These profiles offer valuable insights into customer behaviour and preferences, which are then utilised for marketing and advertising purposes. However, such extensive data collection and analysis raise privacy concerns. A study by Preibusch *et al.* [36] found that over half of 881 online shopping sites shared customer information (names, emails, phone numbers, and purchase details) with PayPal, who then forwarded it to Omniture, a third-party data analytics company. Similarly, Ali Pay, a popular secure payment service for Taobao, has faced accusations of unethical data collection practices [24]. While some argue that analysing customer data can benefit both consumers and merchants, the growing public concern over user privacy has made TPSPs to start thinking about privacy preservation for their consumers.

The architecture of privacy-preserving mobile payment schemes can be either centralised or decentralised. Blockchain's emergence spurred the development of decentralised payment schemes that employ advanced cryptographic techniques to safeguard user privacy. The first decentralised digital currency, Bitcoin, was introduced to enable online transactions without the involvement of a bank or a TPSP [34]. Furthermore, Miers et al. [33] introduced Zerocoin, a cryptographic protocol designed to enhance privacy in cryptocurrencies like Bitcoin by allowing users to convert their bitcoins into untraceable zerocoins. Moreover, Lin et.al [30] proposed a decentralised conditional anonymous payment scheme which protects the privacy of honest users and allows a trusted third party to trace a malicious user's real identity. Despite the theoretical advantage of transaction anonymity offered by their decentralised nature, blockchain-based payment schemes suffer from several critical drawbacks that limit their practicality as a mobile payment solution. Many popular blockchains have a relatively low throughput in terms of the number of transactions they can process per second. This creates a bottleneck, hindering the ability to scale up for the mass volume of transactions in a mobile payment scheme [37]. Blockchain transactions require miners or other participants for validation and security, leading to transaction fees as compensation for their work. These transaction fees are determined by supply and demand. High transaction fees, driven by high demand during network congestion, can make blockchain impractical for small everyday purchases [11, 46]. In addition, the volatility of blockchain-based currencies makes it difficult for merchants to set a stable price for their goods and services [23]. While users are not directly identified with real-world identities, all transactions on the blockchain are publicly visible. With sophisticated analysis, it's possible to link transactions to specific wallets and potentially deanonymise users [17, 21]. Finally, blockchain's decentralised and unregulated nature, while often touted as a strength, paradoxically increases its attractiveness for illicit activities. Studies have shown that the dark web, which heavily uses cryptocurrencies for transactions, has seen a surge in activities such as drug and human trafficking [2, 12]. The difficulty in tracing these transactions back to their origins provides a sense of freedom to those engaging in these illegal operations.

The limitations of decentralised payment schemes mentioned above highlight the need for a regulated intermediary and the importance of privacy-preserving centralised payment schemes. However, ensuring complete user privacy protection within a centralised scheme is also challenging.

Many proposed centralised privacy-preserving payment schemes mainly aim to protect privacy of customers from merchants while allowing the intermediaries (like a bank or TPSP or CA) to access transaction and user-related information [4, 13, 26, 28, 29, 39, 42]. These schemes inherently assume that TPSP/bank/CA will act honestly and not collude with each other to compromise a customer's privacy. Some schemes [4, 13] also rely on the implicit assumption that the merchant will act honestly and will not get involved in any dishonest actions. In reality, this might not be always the case. Even though collecting transaction-related information enables the intermediaries to identify and prevent fraudulent activities performed by the customers such as double-spending, it also allows them to gain sensitive information regarding a customer's spending patterns. Some works have proposed using certificate authorities and virtual accounts to prevent TPSPs from accessing transaction-related information [27, 35, 38]. Implementing and managing such schemes with proxies or virtual accounts can be complex, potentially leading to higher fees or slower transaction processing times. Additionally, these schemes do not offer any guarantee that these proxies are trustworthy and will not collude with the TPSPs.

1.1 Motivation

Our proposal for an untraceable payment system is driven by the limitations we have identified within the existing literature. Firstly, the existing untraceable payment systems prioritise customer privacy under the assumption that key entities like the TPSP or bank or central authority (CA) are trustworthy and will not try to track the transactions. Further, these schemes also assume that these entities will not collude with each other to trace a customer's transaction. Such an assumption may not always hold in real-world scenarios. Our proposed system aims to address this critical gap by explicitly considering the possibility of collusion between these entities. We achieve this without compromising other critical security and privacy requirements. The most important aspect of designing such an untraceable payment scheme lies in how to achieve the following goals simultaneously: 1. Making transactions truly untraceable on the face of curious TPSP while satisfying the necessity for regulatory oversight and ensuring that transaction details remain private, even if participants in the scheme collude. 2. Preventing fraudulent activities such as double-spending and illegal forging of payment tokens. 3. Using a single payment token instead of multiple tokens for different face values. 4. Protecting the messages transmitted between participants from unauthorised access. 5. Ensuring that only authorised participants can initiate messages, and they can confirm each other's identities without revealing any information that could compromise their privacy or enable future tracking. 6. A dishonest customer/merchant should be traceable. 7. Computationally efficient.

Secondly, the pursuit of proposing privacy-preserving untraceable payment systems has been the subject of extensive research which has led to a plethora of proposed schemes. However, in the literature, we can notice that the newly proposed schemes tend to discredit existing solutions and then introduce a "new and improved" payment scheme while neglecting to thoroughly analyse the security and privacy vulnerabilities inherent in their design. This continuous "break-fix-break-fix" cycle [43] has significant consequences. Many new schemes lack robust privacy and security justification leaving them susceptible to potential attacks. This challenge is caused by the absence of a formally defined set of evaluation criteria for assessment. While most works do define their own security and privacy evaluation criteria, they always overlook some other criteria which hinders the understanding of whether the scheme completely addresses the gap we mentioned above. The lack of an appropriate evaluation framework makes it difficult to identify a scheme that truly satisfies all desired criteria. Hence, in this paper, we also propose an evaluation framework that combines the most important privacy and security criteria that should be addressed by an untraceable payment scheme.

1.2 Contributions

Our key contributions are listed below:

- Privacy-Preserving TPSP Integration: We deviate from traditional models where TPSPs have unrestricted access to transaction and customer data. Our proposed scheme introduces a novel protocol which employs blind and partially blind signatures in the registration and token withdrawal stage to achieve complete user privacy protection while meeting the regulatory requirements. We achieve this goal even without relying on any external third parties, streamlining the system architecture and reducing potential collusion-related vulnerabilities.
- Zero-Knowledge Mutual Authentication: We deviate from conventional TPSP-oriented authentication models that risk exposing customer-merchant interactions. Our proposed scheme introduces a mutual authentication scheme built upon a lightweight zero-knowledge proof which is directly between the customer and merchant and adapted for mutual authentication. This protocol uses discrete logarithm-based identity keys generated by the users themselves. However, these keys are verified by the TPSP via blind signature ensuring that even the TPSP cannot link these identity keys to specific users. This approach effectively enables the mutual authentication process between the customer and the merchant, eliminating the TPSP as a potential point of surveillance. This mutual authentication scheme also protects the honest participants from dishonest merchants who try to double-spend.
- Proactive Fraud Prevention: Our proposed scheme enables the TPSP to detect and trace double-spending attempts committed by customers using a novel security tag mechanism, without requiring access to their transaction details. Hence, we continue to maintain the untraceability of legitimate transactions. Additionally, our scheme also upheld the unforgeability property, ensuring that payment tokens cannot be fraudulently created or replicated by malicious adversaries or dishonest customers/merchants. Moreover, our scheme incorporates mechanisms that enable the TPSP to trace the identities of dishonest customers attempting to engage in double-spending, while preserving the anonymity of honest participants.
- Single Token Flexibility: Our scheme also employs a single payment token design that accommodates transactions of varying face values. This is achieved through partially blind signature that allows for the representation of different denominations within a unified token framework.
- Comprehensive Evaluation Framework: We introduce an evaluation framework that serves as a valuable tool for assessing the security and privacy attributes of centralised untraceable payment systems. This framework provides a structured methodology to systematically analyse and compare different protocols, promoting transparency and facilitating informed decision-making during the development and adoption of privacy-preserving payment solutions.

The structure of the remaining portion of this paper is as follows: Section 2 discusses the existing works. Section 3 describes the generic architecture of a third-party-based payment scheme. Section 4 describes the trustworthy and untraceable payment scheme we have proposed in this paper. Section 5 presents the threat model for our proposed scheme and the evaluation criteria that can be used to evaluate the effectiveness of untraceable payment schemes. Section 6 presents the formal security analysis for the proposed scheme and section 7 explains the symbolic modelling process used to evaluate our protocol. Section 8 presents the comparison of our proposed scheme with other schemes. Finally, Section 9 concludes the paper.

2 RELATED WORK

The untraceable centralised payment schemes are classified into online and offline schemes based on the interaction between the customer, merchant, and service provider (either a TPSP or a bank or CA). In online untraceable payment schemes [29, 42], there is a need for the service provider to verify the payment token's validity before the customer can use it to pay a merchant. This requires the service provider to remain constantly online during each transaction, increasing complexity and introducing a potential single point of failure. This questions their suitability for micro-payments in real-world scenarios. Moreover, these schemes assume that the service provider is trustworthy and can provide privacy guarantees against malicious third-party adversaries or dishonest merchants. Recently [7] attempted to address some of these trust and efficiency concerns in online payment schemes. While their scheme tries to protect the privacy of a customer against a curious TPSP, it also assumes that the TPSP will not initiate active attacks to violate the privacy of the customer or collude with other participants like CA to collect transaction-related information. Therefore, addressing the persistent issue of collusion in online payment systems remains crucial for ensuring user privacy.

On the other hand, offline payment systems [4, 8, 20, 26, 39] doesn't require the TPSP to stay online continuously, and the merchant can verify the payment token without the assistance of the TPSP. These works have used numerous methods to reduce the involvement of the TPSP during an active transaction. Some [4, 20] have explored the use of blind signature schemes to reduce the involvement of TPSP and to provide untraceability. These schemes enable customers to have their payment tokens verified by a service provider without revealing any information about the tokens. However, since the customer cannot disclose any information, multiple payment tokens must be issued to differentiate between different face values. As a result, these schemes incur high computational and storage complexity on the customer side. Some works [19, 25] have proposed the use of partially blind signatures [1] to tackle this issue. In partially blind signatures, the customer reveals some information to the service provider and it allows the TPSP to issue a single token for the required value.

However, these schemes have failed to address how a customer and merchant will be able to prove their identity to each other without the involvement of the TPSP. Some offline payment schemes [26, 39] address this issue by assigning anonymous identities to their customers, either issued by the service provider themselves or a third-party anonymous identity provider. This approach helps the customer to prove that they are a legitimate user of the system without revealing their true identity, reducing the risk of revealing any information about the transaction to the TPSP. However, in the case where the TPSP issues these anonymous identities, the TPSP still retains the capability to track transaction details if they decide to act dishonestly. While a third-party anonymous identity provider can enhance privacy compared to TPSP-issued anonymous identities, it introduces more complexity to the existing payment architecture and still these schemes assume that these entities will not collude with the TPSP. Hence, in our proposed payment scheme we employ a RSA-based partially blind signature to completely limit the involvement of the TPSP during token issuance phase. Using the partially blind signature customers can obtain a single payment token for any desired value from the TPSP without revealing any other information about the transaction. Additionally, we have introduced mutual authentication to eliminate the need for anonymous identities, allowing both customers and merchants to confidently verify each other's identity.

A primary concern that arises when minimising the role of the TPSP is the increased risk of double-spending attacks. The Cut and Choose technique [44] has been used in some schemes to address the double-spending problem caused by the customer. However, this method incurs

additional computational and communication overhead. Some works have introduced [18, 19] restrictive blind signatures to prevent double-spending, where the customer can blind the message body but not its internal structure. If the customer attempts to double-spend the payment token, the TPSP can identify the dishonest customer through the message structure. This approach relies on the TPSP's trustworthiness, assuming they will only disclose transaction details in cases of double-spending, not for routine tracking. Hence, in our proposed scheme we introduce a security tag mechanism that enables the TPSP to detect and trace double-spending attempts by the customer. This mechanism is designed in such a way that it prevents the TPSP from revealing any transaction details unless double-spending has occurred. In summary, we propose a centralised payment scheme that achieves strong user privacy protection, robust fraud prevention, and efficient implementation without relying on complex third-party mechanisms.

3 ARCHITECTURE OF A GENERIC TPSP BASED MOBILE PAYMENT SCHEMES

This section provides an overview of the generic TPSP-based payment scheme. In the generic scheme, the TPSP collects all transaction-related information of a customer. For a visual breakdown of how a typical purchase happens, refer to Fig. 1. The transaction between a customer and a merchant in the generic scheme consists of the following stages:

- Tokenising Card: When a customer first signs up with the TPSP, they add their credit or debit card details. To keep this sensitive information safe, the scheme creates a unique code called a "token" to replace it. This token is stored on the customer's phone for contactless payments (NFC) and the TPSP also keeps a copy for reference.
- Paying the merchant: When the customer wants to buy something, they send the order details to the merchant, who confirms the order and the total cost. Once the customer receives confirmation of the order total from the merchant, they can pay by tapping their phone on the merchant's payment terminal.
- Merchant payment processing: The merchant's terminal then uses the token details received from the customer's phone to process the payment through the TPSP.
- TPSP translating the payment token: The TPSP first verifies the payment token details provided by the merchant. If the validation is successful, the TPSP translates the token information back into the actual card number and uses it to identify the customer's account. The relevant amount is then deducted from the customer's account and temporarily placed into a holding account. Simultaneously, the TPSP sends an authorisation response to the point-of-sale terminal, confirming whether or not the transaction was successful.
- Terminal notification: The point of sale terminal notifies both the merchant and the customer whether or not the payment attempt is successful.

4 PROPOSED UNTRACEABLE PAYMENT PROTOCOL SCHEME

4.1 Overview of the Protocol

Our proposed untraceable payment scheme involves three parties: a customer (C), a merchant (M), and a third-party payment service provider ($TPSP$). It comprises of six stages:

- User Registration: Both customers and merchants need to register with the TPSP to use the scheme.
- Purchase Request: When a customer wants to buy something, they first establish a secure connection with the merchant. Even though the customer and merchant are registered with the TPSP, the goal of an untraceable scheme is to limit the TPSP's knowledge of individual transactions. Hence, the customer and the merchant will use a mutual authentication scheme to verify each other.

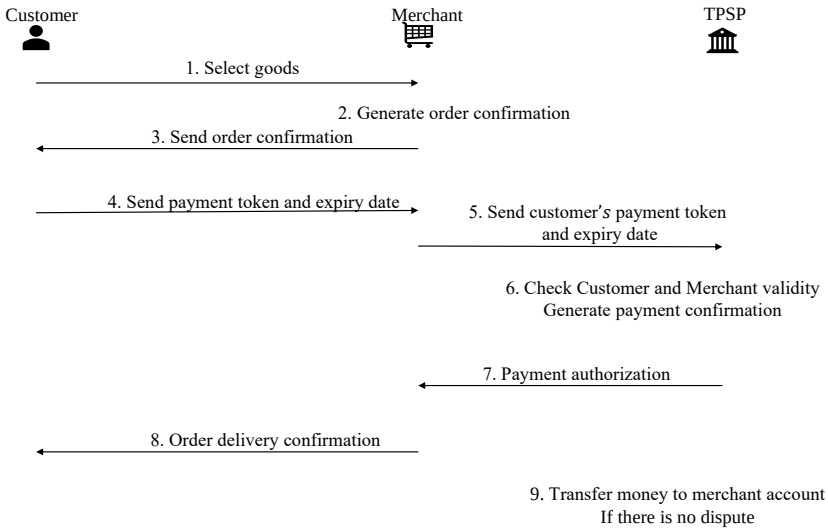

Fig. 1. Workflow of the Generic TPSP based Mobile Payment Scheme

- Token Withdrawal: Before making the payment, the customer gets a unique payment token of the required value from the TPSP. In our scheme, we ensure the customer is not re-identified by the TPSP using this token.
- Payment Using Token: The customer then sends the payment token obtained from the TPSP to the merchant.
- Token Deposit: The merchant verifies the token and then forwards the token to the TPSP, who checks if it's valid. If it is a valid token then the TPSP authorises the payment.
- Refund: If the customer isn't happy with the product or service, there's an optional refund process.

Fig. 2 displays the primary stages involved in the proposed untraceable payment scheme. The key differences between the generic TPSP based payment scheme and our proposed untraceable payment scheme can be summarised as follows:

(1) TPSP's Role: In the generic scheme, TPSP plays an active role throughout the transaction. It stores all information related to a transaction, issues/validates payment tokens and verifies the validity of customers' and merchants' identities. Whereas in our proposed scheme, TPSP handles user registration and issuance/validation of blinded payment tokens. They do not aggregate/store information regarding the transactions, and the customer and the merchant can verify their identities without the involvement of the TPSP.
(2) Complete User Privacy: In the generic scheme, TPSP collects and stores a significant amount of user data including transaction details. This allows the TPSP to build detailed profiles related to a customer's preferences which raises privacy concerns. On the other hand, our proposed scheme prioritises user privacy. The TPSP's data collection is limited to essential information like card details and transaction amounts. We use zero knowledge-based mutual

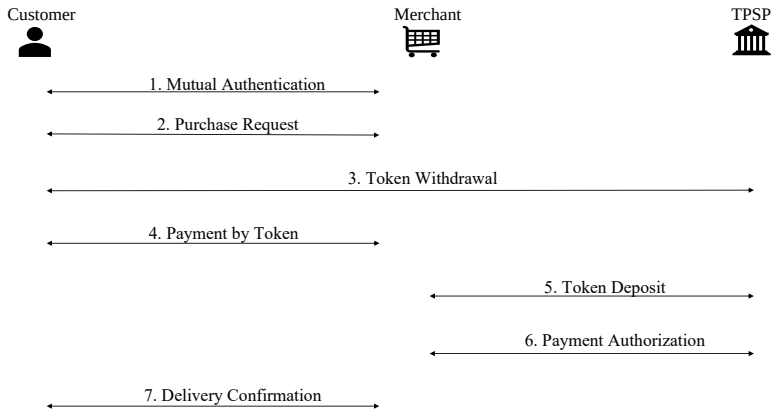

Fig. 2. Workflow of the Proposed Untraceable Mobile Payment Scheme

authentication scheme and partially blind signatures to ensure that the TPSP cannot track individual purchases to build comprehensive customer profiles.

(3) Authentication: The authentication process in the generic scheme involves the TPSP verifying both the customer and merchant identities at every stage. Our scheme utilises a zero-knowledge proof-based mutual authentication mechanism. This allows the customer and the merchant to verify each other's identity directly. However, we ensure the customer and merchant are registered valid users who are authorised to use the platform.

(4) Double-Spending Prevention: The mechanism for preventing double-spending in the generic scheme relies on the TPSP's ability to track transactions. We, on the other hand, have implemented appropriate mechanisms to detect and prevent double-spending attempts carried out by both customers and merchants. The TPSP can trace the origin of a double-spent token, even though individual transactions are untraceable.

In essence, our proposed scheme aims to address the privacy concerns associated with traditional TPSP-based mobile payment schemes by significantly reducing the TPSP's access to user data and transaction details.

4.2 Preliminaries

Our proposed payment scheme utilises several cryptographic primitives to preserve customer's privacy and maintain security. This section provides a brief overview of these technologies.

4.2.1 Zero Knowledge Proofs. Zero-knowledge proofs (ZKPs) allow one party (the prover) to assure another party (the verifier) that they know a secret without revealing the secret itself. The prover who possesses the secret constructs a series of mathematical proofs that are tied to this secret. These proofs which often involve modular arithmetic, elliptic curve cryptography, or other advanced mathematical constructs, are meticulously crafted to demonstrate possession of the secret without revealing its actual value. The verifier subjects these proofs to validation which involves a series of computations and verification against publicly known parameters. The strength of a ZKP protocol depends on its ability to ensure that a successful validation by the verifier clearly implies the prover's possession of the secret, even though the verifier gains no information regarding the secret itself. In the context of our proposed payment scheme, ZKPs

enable mutual authentication between the customer and merchant, ensuring they are legitimate entities without disclosing identifying information to the TPSP or to each other. ZKP was first introduced by [14] as an efficient cryptographic primitive that requires less computational power compared to other public-key-based authentication methods. Motivated by this [31] proposed a lightweight ZKP scheme to manage the authentication process in wireless body area networks. However, this scheme relies on the base station to manage the key verification. Hence, in this paper we have made substantial changes to the scheme proposed by [31] to make it suitable for a mutual authentication scheme with limited TPSP involvement.

4.2.2 Partially Blind Signature. The partially blind signature acts as a specialised digital signature scheme where the signer (TPSP in our context) validates a message without having full visibility of its content. In our payment scheme the message, i.e. the payment token, is first 'blinded' by the customer using a random blinding factor. This blinding process hides the details of the payment token so that the TPSP cannot link it back to a specific customer when they use it. However, unlike blind signatures, partially blind signatures allow certain pre-agreed information such as the transaction amount to be visible to the TPSP. The TPSP then signs the blinded token, and the customer will be able to obtain a valid signature on the original payment token. Partially blind signature allows us to incorporate two seemingly conflicting objectives: user privacy and regulatory oversight. The blinding process is used to safeguard the customer's privacy by preventing the TPSP from tracking individual transactions through payment tokens. Simultaneously, since the TPSP has access to some information (value of payment token withdrawn), it allows the TPSP to fulfil its regulatory obligations, such as monitoring transaction amounts for potential fraud or money laundering activities.

4.3 Registration and Key Generation

Both the customer and the merchant must first register with the TPSP so that the TPSP can generate an RSA key pair for each registered user. Each user receives a public key (P_{key}) which they can share and a private key (S_{key}) which they keep secret. Additionally, each user generates their own long-term identity key pairs, consisting of public identity keys ($V_{m,j}$) which they can share and private identity keys ($S_{m,j}$) which are stored locally and kept secret. These keys help verify the customer's identity during transactions with merchants without revealing personal information. While users never disclose their identity keys to the TPSP, they do share their public identity keys ($V_{m,j}$) for validation. However, these keys are blinded using a blind signature proposed by Zhang and Kim [45] before sharing with the TPSP. The blinding process prevents the TPSP from linking a user's identity to the specific public identity key pair. On the other hand, the validation process ensures the keys used during transactions are legitimate and belong to registered users. Algorithm 1 shows the steps involved in the registration and the key generation stage. The detailed registration and key generation process is detailed below:

- TPSP chooses two large prime numbers p_z and q_z and computes $N = p_z \times q_z$. The prime numbers p_z and q_z are kept private and N is a public parameter.
- Each registered user generates k number of public and private identity key pairs locally on their device where $k = 1, 2, ..., 20$. We use $V_{m,j}$ and $S_{m,j}$ to represent the group of public identity keys and private identity keys generated by the user m respectively. The public and private key generation is as follows:
  - Generation of private keys: Randomly generate an integer $S_{m,j} = S_{m,1} - j + 1$ $(2 \leq j \leq k)$ such that $1 \leq S_{m,1} \leq N - 1$.
  - Generation of public keys: $V_{m,j} = \frac{1}{S_{m,j}^2} mod\ N$ $(1 \leq j \leq k)$.

- Let P be the generator of an additive cyclic group $G1$ of prime order q. $G2$ is a multiplicative cyclic group with the same order q. The bi-linear pairing e is defined as $e : G1 \times G1 \rightarrow G2$. The TPSP selects a random number s as the master secret key, which remains confidential to the TPSP. The TPSP then sets a system parameter $P_{pub} = sP$. Subsequently, the TPSP chooses two hash functions, namely $H1$ and $H2$. The TPSP then shares the system parameters $G1, G2, e, q, P, P_{pub}, H1$ and $H2$ with each registered user and keeps s private.
- When a user submits their identity information (ID) to the TPSP, the TPSP then generates a public key $Q_{ID} = H2(ID)$ and private key $S_{ID} = sQ_{ID}$.
- TPSP chooses a random number $r \in \mathbb{Z}_{\shortparallel}$ and computes a commitment value $U = rQ_{ID}$. The TPSP then sends U to the user.
- The user then chooses blinding factors $\alpha, \beta \in \mathbb{Z}_{\shortparallel}$ and computes $U' = \alpha U + \alpha\beta Q_{ID}$. Then the user blinds the public identity keys $V_{m,j}$ by computing a blind message $h = \alpha^{-1}H1(V_{m,j}, U')+\beta$. The blind message h is then sent to the TPSP for validation.
- TPSP computes $V = (r + h)S_{ID}$ and sends V back to the user.
- User computes $V' = \alpha V$. The TPSP's signature on the public identity keys $V_{m,j}$ is (U', V').

Algorithm 1 Untraceable Payment Scheme - Registration and Key Generation Stage

TPSP (For all users):
for each user **do**
    Generate RSA key pair
    Set system parameters
end for
Each user:
    Generate identity key pairs for themselves
    Blind public identity keys using Zhang and Kim blind signature scheme [45]
    Send blinded keys to TPSP for validation
    Receive validation response from TPSP
    Store validated keys securely

Users can verify the legitimacy of another user's public identity key $V_{m,j}$ using the public system parameters distributed by the TPSP and Q_{ID}. The signature is acknowledged only if $e(V', P) = e(U' + H1(V_{m,j}, U')Q_{ID}, P_{pub})$.

4.4 ZKP Mutual Authentication

Registered user in our payment scheme will use a commonly used authentication method when proving their identity to the TPSP. However, when the customers and the merchants want to initiate a transaction in our payment scheme, they use a zero knowledge-based mutual authentication scheme without needing any assistance from the TPSP. Our proposed scheme allows customers and merchants to switch roles during this mutual authentication phase. When a party proves their identity as a "prover," the other party verifies their credentials as a "verifier." Our proposed mutual authentication scheme guarantees the verifier cannot learn the prover's private identity key, but can still confirm that the prover is a registered user. Additionally, this mutual authentication scheme also prevents merchants from double-spending the token as they never learn the customer's private key. Hence, they cannot impersonate a valid customer to make fraudulent purchases with other merchants. Algorithm 2 shows the steps involved in the mutual authentication process. The mutual authentication scheme is described in detail below:

- TPSP uses the parameters generated at the registration stage p_z, q_z and N in the mutual authentication scheme.
- The prover initially sends an authentication request message to the verifier requesting an authentication challenge.
- The verifier then generates a challenge $M_{chall} : (e_1, e_2, ...e_k)$ where $e_k = 0 \; or \; 1$ and $k = 1, 2, ..., 20$. The verifier then sends the challenge M_{chall} and a timestamp T_1 to the prover.
- After receiving the challenge, the prover selects a random number a such that $1 \leq a \leq N-1$ and computes $X_m = a^2 \; mod \; N$ and $Y_m = a \prod_{j=1}^{k} S_{m,j}^{e_j} \; mod \; N \; (1 \leq j \leq k)$ where $S_{m,j}$ are the private identity keys of user m.
- The hashed value $H(X_m)$ is computed using SHA-1 algorithm and the prover will send $H(X_m)$, Y_m, public identity keys $V_{m,j}$, Q_{ID} along with the TPSP's signature (U', V') and a timestamp T_2 to the verifier.
- The verifier initially verifies whether $\Delta t > T$ where $\Delta t = T_2 - T_1$ and T is the response time threshold set by the verifier. If this verification fails, then the verifier rejects the authentication reply from the prover.
- If $\Delta t <= T$, then the verifier verifies whether the signature (U', V') is valid by performing a verification check using the system parameters issued by the TPSP as explained in section 4.3. If the signature validity verification fails, then the verifier rejects the authentication reply from the prover.
- If the signature verification is successful, then the verifier computes X'_m using the provers' identity public key as $X'_m = Y_m^2 \prod_{j=1}^{k} V_{m,j}^{e_j} \; mod \; N \; (1 \leq j \leq k)$. If X'_m is equivalent to X_m then the prover will be authenticated successfully by the verifier.

The payment scheme can proceed to the next step of initiating a purchase request if and only if the mutual authentication process is successful.

Algorithm 2 Untraceable Payment Scheme - ZKP Mutual Authentication

Prover:
Send authentication request to the verifier
Verifier:
Generate a ZKP challenge and timestamp.
Send the challenge to the prover.
Prover:
Compute the answer to the challenge to prove their identity using their secret keys.
Send the answer to the verifier along with the TPSP's signature to the verifier.
Verifier:
Verifies whether the signature is actually from the TPSP
if Verification is successful **then**:
Authentication is successful.
else:
Authentication failed.
end if

4.5 Payment Process

This phase consists of five stages: purchase request, token withdrawal, token issuance, payment by token and token deposit. Algorithm 3 provides a detailed breakdown of the steps involved in each stage of the transaction process.

Algorithm 3 Untraceable Payment Scheme - Transaction Process

Stage 1: Purchase Request
Customer
Sends request to buy specific products to Merchant
Merchant
Generates security tag, total amount, and a unique transaction ID
Sends price and transaction details back to Customer
Stage 2: Token Withdrawal
TPSP
Generates parameters for partially blind signature
Publishes necessary public information to all their users
Customer
Generates unique token ID and random numbers needed for the process
Blinds the token ID so that TPSP will not be able to see it
Sends blinded token ID and corresponding total value of the transaction to the TPSP
Customer and TPSP exchange random values required for token issuance stage
Stage 3: Token Issuance
TPSP
Generates a signature on blinded token without seeing the token ID
Sends the required signature related values to the Customer
Customer
Unblinds the signature to get a valid signature on the token ID
Stage 4: Payment By Token
Customer
Generates a security tag to help detect double-spending
Sends the token ID, signature, and security tag to Merchant
Stage 5: Token Deposit
Merchant
Verifies the signature on the token
If signature is valid, send token ID, signature and security tag to the TPSP
TPSP
Verifies the signature on the token

if Verification is successful **then**:
TPSP informs the Merchant and processes the deposit;
TPSP stores the token ID and security tag;
else:
TPSP sends transaction failure message to Merchant;
TPSP initiates double-spending detection;
end if

4.5.1 Purchase Request. In the purchase request stage, the customer sends a purchase request to the merchant. The purchase request stage can be described in detail as follows:

- First C sends a purchase request containing the list of product IDs $(Item_1, Item_2, ...Item_n)$ to M.
- M generates a random number $R \in \mathbb{Z}_{||}$, accumulates the prices for the list of products X and generates a transaction ID P_{ID}.
- M sends a purchase reply tuple (R, P_{ID}, X) to C.

We use the random number R to generate a security tag crucial for double-spending detection.

4.5.2 Token Withdrawal. Our scheme employs an RSA-based partially blind signature [9] to reduce the computational complexity. We use the product price (X) as the information that can be shared with the TPSP. Before withdrawing the token, the TPSP selects two large prime numbers p_t and q_t. Then the TPSP calculates a modulus $n_t = p_t \cdot q_t$. In addition, the TPSP also computes the totient function $\phi(n_t) = (p_t - 1) \cdot (q_t - 1)$ and private exponent d_t such that $e_t \cdot d_t \equiv 1 \mod \phi(n_t)$, where e_t is the blinding factor. The TPSP discloses (e_t, n_t) and keeps the values of (d_t, p_t, q_t) private. Furthermore, the TPSP publishes a hashing algorithm $H3$. The detailed steps of the token withdrawal stage are listed below:

- C generates a random Token ID T_{ID} and randomly chooses two numbers r_t and u_t where $r_t, u_t \in \mathbf{Z_{n_t}}$.
- C blinds the Token ID T_{ID} and creates the blinded token ID $Y = r_t^{e_t} H3(T_{ID})(u_t^2 + 1) \ mod \ n_t$, where Y is the resulting message.
- C then sends (X, Y) to the TPSP.
- TPSP randomly chooses a positive integer a_t where $a_t < n_t$ and sends it to C.
- After receiving a_t, C randomly generates another random number r'_t and computes $b_t = r_t.r'_t$.
- C then computes $\beta = b_t^{e_t}(u_t - a_t) \ mod \ n_t$ and sends β to the TPSP.

4.5.3 Token Issuance. In the token issuance stage, the TPSP issues a valid digital signature on the payment token without seeing or storing the Token ID. The process is described as follows:

- After receiving β from C, TPSP computes $\beta^{-1} \ mod \ n_t$ and computes $t = H3(X)^{d_t}(Y(a_t^2 + 1)\beta^{-2})^{2d_t} \ mod \ n_t$.
- TPSP then sends (β^{-1}, t) to C.
- Upon receiving (β^{-1}, t), C acquires the signature of TPSP on Token ID T_{ID} by computing $c = (u_t a_t + 1).\beta^{-1}.b_t^{e_t} \ mod \ n_t$, and $s = t.r_t^2.r_t'^4 \ mod \ n_t$. C uses (X, c, s) as the signature on Token ID T_{ID}.

4.5.4 Payment By Token. Following successful validation of the payment token, the customer C submits the token to the merchant M. As part of the double-spending detection mechanism, customer C also generates a unique security tag. The security tag is computed using the customer's public (P_{key}) and private (S_{key}) keys assigned during the registration stage by the TPSP. The process for payment by the token is detailed below:

- C generates a security tag $T_c = P_{key}^{(1+R)}$ using the random number R sent by M in the purchase request stage and the public key P_{key}.
- C then sends a payment request tuple (T_{ID}, X, T_c, c, s) to M.

4.5.5 Token Deposit. Merchant M verifies the signature on the Token ID, and then forwards the payment tuple to TPSP to initiate token deposit. The token deposit stage process is described below:

- M verifies whether the signature on the payment token is valid by computing $s^{e_t} = H3(X)H3(T_{ID})^2(c^2+1)^2 \; mod \; n_t$. If the verification fails, then M sends **Payment Failed** message to C.
- If the verification is successful, M forwards the payment request tuple (T_{ID}, X, T_c, c, s) to TPSP and at the same time stores T_c, R, T_{ID} locally.
- TPSP verifies whether the signature on the payment token is valid. If the verification fails, then TPSP sends **Token Deposit Failed** message to M.
- Upon successful verification, the TPSP verifies whether the Token ID (T_{ID}) already exists in the token deposit database. If the token ID is found, the TPSP sends a **Token Deposit Failed** message to the merchant (M) and initiates the double-spending detection process.
- If the verification is successful, TPSP sends **Token Deposit Successful** message to M.

Following a successful token deposit, the TPSP adds the Token ID to the token deposit database. The TPSP then sends a payment confirmation message to the merchant (M). Upon receiving confirmation, M sends the product to the customer and transmits a product delivery confirmation, concluding the transaction between the customer (C) and the merchant (M).

4.6 Refunding

Our protocol incorporates a customer-initiated refund mechanism, allowing customers to request a refund directly from the merchant, bypassing the TPSP. During the refund process, the roles of the customer and merchant are reversed in contrast to the transaction process. The merchant assumes responsibility for executing the token withdrawal, token issuance, and payment by token phases with the TPSP to obtain a valid anonymous token with the appropriate denomination for the refund. Conversely, the customer is responsible for the final step - the token deposit - by submitting the payment tuple to the TPSP. Once the refund token is successfully deposited, the TPSP stores the corresponding refund Token ID in the token deposit database.

5 THREAT MODEL AND EVALUATION CRITERIA

5.1 Threat Model and the Adversary

We consider two types of adversaries when it comes to our proposed scheme. Firstly we assume Dolev-Yao adversary [10] based external threat model. Such type of an adversary is assumed to possess capabilities such as eavesdropping, modifying, intercepting, injecting and deleting any messages transmitted between the customer, merchant, and the TPSP on public communication channels. However, we assume that the power of the adversary is restricted by the assumption of perfect cryptography. Hence they cannot break the encryption or digital signature schemes used in our protocol to gain unauthorised access to information or forge messages that appear to be coming from legitimate participants. The adversary is powerful when it comes to controlling the public communication channel but is limited by the strength of the cryptographic primitives.

In addition to the Dolev-Yao attacker, we also consider the possibility of insider attacks conducted by dishonest participants within the protocol. Firstly we assume a dishonest customer who will attempt to double-spend their payment token [5] by reusing the same token they used to pay for another purchase elsewhere. Secondly, we assume a dishonest merchant [3] whose goal is to double-spend the token they received from the customer on another transaction. We consider the possibility of this dishonest customer/merchant attempting to spend the same token multiple times. We assume such an adversary cannot directly manipulate the TPSP's internal records or gain access to other users' accounts. We also assume another type of adversary, a fraudulent

customer/merchant, who may resort to token forgery [15] to gain unauthorised access to goods or services. Most importantly we assume curious TPSP [7] is very harmful to our proposed scheme. While we assume that the TPSP is honest and will not perform any dishonest activities such as impersonating a customer, they might exhibit curiosity and attempt to analyse transactional data to link customers with their purchases. Even without malicious intent, a TPSP's curiosity can lead to a significant privacy breach. Such privacy breaches can severely erode trust in the system in return.

In this paper, we do not address an adversary who may compromise either the customer's or the merchant's device, gaining access to their secret keys and then gaining the ability to initiate transactions on their behalf. This type of attack, where the adversary gains control of user credentials, aligns with the capabilities of an adversary as defined within the Camenisch-Lysyanskaya model [6]. If a merchant device is compromised, the adversary can impersonate the merchant and obtain valid tokens from the customer and then refuse to offer the goods/services. However, our scheme ensures that they cannot double-spend the token. Additionally, the scheme doesn't protect against any attacks when the customer device is compromised and the adversary can continue to initiate a new transaction. The ZKP-based mutual authentication scheme does not stop the adversary from creating new authentication responses if they have access to the customer's secret keys. To mitigate this risk, the TPSP can impose spending limits on individual transactions or on a daily/monthly basis. Moreover, customers should be encouraged to promptly inform the TPSP if their device is lost or stolen, allowing the TPSP to take necessary actions such as blocking the associated account or revoking the compromised keys to prevent any further unauthorised transactions.

5.2 Evaluation Criteria

Currently, there's no universally accepted standard for the security and privacy features that an untraceable payment scheme should consists of. Hanatani et.al [15] established four security goals for untraceable payment systems. However, their framework doesn't fully address the risk of curious TPSP. Several other works [1, 4, 6, 26, 28] have proposed their evaluation criteria, but these are often inconsistent and partial to their own work. Recently, [16] proposed ten desirable security goals for decentralised payment systems. Building on this work, we propose a set of privacy and security criteria that any centralised untraceable, TPSP-owned payment system should meet. Table 1 shows the criteria we have proposed in this paper.

Most previous schemes [4, 7, 8, 26, 28, 29, 39, 42] ignore possibility of collusion within the payment scheme. Hence, we follow the definition by [15] and re-define the evaluation criteria "C1" for the untraceable payment schemes. Furthermore, we further enhance the definition of exculpability ("C2") compared to other works. Rather than solely focusing on dishonest customers [15], our criteria require payment schemes to prevent double-spending attempts by both customers and merchants. We use the same definition for criteria "C3, C4" and "C5" as given in [26]. In addition, we introduce another criterion "C7" to strike a balance between preventing illicit use of the untraceable payment scheme and safeguarding customers' privacy. The criteria "C7" requires the untraceable payment scheme to collect necessary information and at the same time prioritise preserving the privacy of users.

Table 1. Evaluation Criteria

C1	Untraceability: The system should ensure that the TPSP or any other entity cannot link a customer to their specific transactions or payment tokens.
C2-a	Exculpability - Merchant Double-spending: The system should have mechanisms to prevent double-spending attempts by the merchants and to trace the dishonest participant.
C2-a	Exculpability - Customer Double-spending: The system should have mechanisms to prevent double-spending attempts by the customers and to trace the dishonest participant.
C3	Unforgeability: The system should guarantee that only authorised entities such as TPSP can create legitimate payment tokens, preventing any attempts to forge tokens.
C4	Confidentiality: Information transmitted between the legitimate participants of the payment scheme should be protected from unauthorised access.
C5	Message Authenticity: The system should ensure that messages and transactions are originating from legitimate participants on the system.
C6	Efficiency: The system should be computationally efficient.
C7	Regulatory Compliance: The system should allow the TPSP to collect necessary information for regulatory purposes while still preserving user privacy.

6 COMPLEXITY-BASED FORMAL SECURITY PROOF

The complexity-based formal proof given in this section ensures the protocol's robustness against powerful internal and external adversaries. Through the proof we demonstrate that the security of the protocol is directly linked to the strength of the RSA assumption which showcases its resilience. The proof also shows that the protocol fulfils its intended goals such as confidentiality, message authenticity, untraceability, unforgeability and exculpability.

6.1 Confidentiality

In the context of our proposed scheme, confidentiality ensures the adversary cannot learn the link between a customer and their payment token or the details of the purchase request. We aim to demonstrate that our protocol adheres to the above definition using the following proof.

Theorem 6.1. *Let π be the untraceable payment protocol described in this paper. Under the RSA assumption, π achieves confidentiality against a probabilistic polynomial-time (PPT) adversary.*

Proof. We will prove this by contradiction. Assume there exists a PPT adversary who can break the confidentiality of the protocol π with non-negligible probability ϵ. We will construct a PPT algorithm AlgoCF that uses the adversary as a subroutine to solve the RSA problem with non-negligible probability. This contradicts the RSA assumption. The steps involved in the AlgoCF algorithm are as follows:

(1) AlgoCF receives an RSA challenge (N, e, y). The objective of the algorithm is to determine the value of x such that it satisfies $x^e \equiv y \pmod{N}$.
(2) AlgoCF simulates the role of the TPSP and engages with the adversary who is allowed to take on the roles of both the customer and the merchant. It is crucial to note that AlgoCF generates all the parameters of the protocol honestly. The exception is the specific modification outlined below.
    - If the adversary is acting as the customer, AlgoCF strategically embeds the RSA challenge y into the protocol's operation. During the token issuance phase, AlgoCF deviates from

the standard computation of $t = H_3(X)^{d_t}(Y(a_t^2+1)\beta^{-2})^{2d_t} \pmod{n_t}$. Instead it computes $t = y^d(Y(a_t^2+1)\beta^{-2})^{2d} \pmod{N}$ where d represents the RSA private key corresponding to the public key (e, N) employed in the partially blind signature scheme. AlgoCF is unaware of the value of d_t.

(3) The adversary proceeds to interact with the simulated protocol π and tries to violate the confidentiality property of the protocol. If the adversary succeeds with the probability ϵ, it means that the adversary possesses the capability to differentiate between two different scenarios, a) The protocol is executed in an honest and unmodified manner or b) During the token issuance process, the hash of the Token ID, denoted as $H3(TID)$, is substituted with the RSA challenge y in the computation of t.

(4) The ability of the adversary to differentiate between these scenarios means that the adversary has acquired some information regarding the blinded Token ID. Hence, AlgoCF can exploit this information to compute the e-th root of $y \pmod{N}$.

If the AlgoCF successfully solves the RSA problem with a non-negligible probability, that outcome directly contradicts the fundamental RSA assumption. Hence, it makes our initial assumption about the existence of such an adversary incorrect. Therefore, we conclude that the proposed untraceable payment protocol π essentially ensures confidentiality under the assumption of the intractability of the RSA problem. □

6.2 Message Authenticity

In the context of the proposed untraceable payment protocol, authenticity is defined as the guarantee that the communication between the parties (customer, merchant, and TPSP) indeed originated by a valid party within the protocol and has not been modified or tampered with during the transmission by an adversary. We aim to demonstrate that our protocol adheres to the above definition using the following proof.

Theorem 6.2. *Let π be the proposed untraceable payment protocol. Under the RSA assumption, π achieves message authenticity against a probabilistic polynomial-time (PPT) adversary.*

Proof. We will prove this by contradiction. Let's assume there exists a PPT adversary who can break the message authenticity of the protocol π with non-negligible probability ϵ. That means the adversary can forge or modify a message in a way that successfully deceives an honest participant with a probability ϵ. The adversary can eavesdrop on all communication channels and can initiate sessions with the participants pretending to be participants in the protocol. We will construct a PPT algorithm AlgoMA that utilises the adversary as a subroutine to solve the RSA problem using the following steps.

(1) AlgoMA receives an RSA challenge tuple (N, e, y) and its goal is to compute the value of x such that $x^e \equiv y(\pmod{N})$. The algorithm simulates the protocol π and embeds the RSA challenge within the protocol execution.

(2) AlgoMA acts as the TPSP and, during the token issuance phase, modifies the computation of t as $t = y^d(Y(a_t^2+1)\beta^{-2})^{2d} \pmod{N}$, where d is the RSA private key corresponding to the public key (e, N) used in the partially blind signature scheme. Note that AlgoMA does not know the value of d_t.

(3) The adversary now tries to violate the message authenticity property of the simulated protocol. A successful attack means the adversary was allowed to create a message that looks like it came from a legitimate participant. There are two types of forgery attempts the adversary can perform:

- Forging a payment token: The adversary can act as a customer and attempt to forge a payment token and the associated signature to deceive a merchant. If the adversary successfully forges a payment token TID^* and its signature (c^*, s^*), the signature verification equation $(s^*)^e \equiv H3(X)H3(TID^*)^2((c^*)^2+1)^2(modN)$ will hold. Since AlgoMA embedded the RSA challenge y in the computation of t during the token issuance phase for TID^*, it knows that $H3(TID^*) = y$. Therefore, $(s*)^e \equiv H3(X)y^2((c*)^2+1)^2(modN)$. AlgoMA can compute the right side of this equation and obtain a value z where $(s^*)^e \equiv z(modN)$. Using the Extended Euclidean Algorithm, AlgoMA can compute $z^{-1}(modN)$ and finally compute $x = s * z^{-1}(modN)$. This x satisfies $x^e \equiv y(modN)$, thus solving the RSA challenge.
- Modifying a message: The adversary could intercept and modify a message exchanged between two participants of the actual protocol. If the adversary successfully modifies a message then AlgoMA can be used to analyse the difference between the original and modified messages. This can potentially reveal information about the secret keys or random values used in the protocol. Hence, AlgoMA can derive the solution to the RSA problem by carefully analysing the revealed information.

If AlgoMA can solve the RSA problem with non-negligible probability, it contradicts the RSA assumption, which states that factoring large RSA moduli is computationally infeasible. Therefore, our initial assumption about the existence of an adversary capable of breaking message authenticity is incorrect. □

6.3 Untraceability

In the context of the protocol proposed in this paper, a payment token is untraceable if for a given unblinded signature (X, c, s) on the token ID TID, if TPSP or merchant cannot link the signature to the identity of the customer who requested the token. We aim to demonstrate that our protocol adheres to the above definition using the following proof.

Theorem 6.3. *The proposed protocol achieves untraceability against a probabilistic polynomial-time (PPT) adversary, assuming the RSA problem is hard.*

Proof. Let's assume that TPSP stores a record of all the values of the parameters it computes and receives from each customer during the token withdrawal and issuance phases. The untraceability property is preserved if $s = t.r_t^2.r_t'^4 \ mod \ n_t$ computed in the Token Issuance stage holds valid for all the signature records stored by the TPSP. Let $\{X, T_{ID}, c, s\}$ be a valid signature tuple, and $\{X, Y, a_t, \beta, t\}$ be the corresponding signature record stored by the TPSP. For the sake of contradiction let's assume that the TPSP can derive b_t, r_t and u_t in such a way they satisfy the following equations from the protocol:

$$Y = r_t^{e_t} H3(T_{ID})(u_t^2 + 1) \ mod \ n_t, \tag{1}$$

$$\beta = b_t^{e_t}(u_t - a_t) \ mod \ n_t \tag{2}$$

$$c = (u_t a_t + 1)(u_t - a_t)^{-1} \ mod \ n_t \tag{3}$$

From Eq. (3) we can we can express u_t as

$$u_t = (ca_t + 1)(c - a_t)^{-1} \ mod \ n_t \tag{4}$$

If we substitute Eq. (4) into Eq. (1), we get

$$Y = r_t^{e_t} H3(T_{ID})(((ca_t + 1)(c - a_t)^{-1})^2 + 1) \ mod \ n_t.$$

From this we can derive r_t as

$$r_t = Y^{d_t} H3(T_{ID})^{-d_t}(u_t^2 + 1)^{-d_t} \ mod \ n_t \tag{5}$$

Similarly, substituting the value of u_t into Eq. (2), we get

$$\beta = b_t^{e_t}((ca_t + 1)(c - a_t)^{-1} - a_t) \; mod \; n_t$$

Then we can derive b_t as

$$b_t = \beta_t^d (u_t - a_t)^{-d_t} \; mod \; n_t \tag{6}$$

Now, even if the TPSP learns the values of b_t, r_t, u_t from the signature record tuple X, Y, a_t, β, t, these values get eliminated when we substitute them back into the original signature verification equation. This is demonstrated by the following reduction:

$$\begin{aligned}
s &= t \cdot r_t^2 \cdot r_t'^4 \bmod n_t \\
&= H_3(X)^{d_t}(Y(a_t^2+1)\beta^{-2})^{2d_t} \cdot r_t^{-2} \cdot b_t^4 \bmod n_t \\
&= H_3(X)^{d_t}(Y(a_t^2+1)\beta^{-2})^{2d_t} \cdot [Y^{d_t} H_3(TID)^{-d_t}(((ca_t+1)(c-a_t)^{-1})^2+1)^{-d_t}]^{-2} \\
&\cdot [\beta^{d_t}[((ca_t+1)(c-a_t)^{-1}) - a_t]^{-d_t}]^4 \bmod n_t \\
&= H_3(X)^{d_t} H_3(TID)^{2d_t}[(a_t^2+1)(u_t^2+1)]^{2d_t}\beta^{-4d_t} \cdot Y^{-2d_t} H_3(TID)^{2d_t}(u_t^2+1)^{-2d_t} \\
&\cdot \beta^{4d_t}[(u_t - a_t)^{-4d_t}] \bmod n_t \\
&= H_3(X)^{d_t} H_3(TID)^{2d_t}(a_t^2+1)^{2d_t} \cdot [r_t^{e_t} H_3(TID)(u_t^2+1)]^{-2d_t}(u_t - a_t)^{-4d_t} \bmod n_t \\
&= H_3(X)^{d_t} H_3(TID)^{2d_t}(a_t^2+1)^{2d_t} \cdot r_t^{-2d_t e_t} H_3(TID)^{-2d_t}(u_t^2+1)^{-2d_t}(u_t - a_t)^{-4d_t} \bmod n_t \\
&= H_3(X)^{d_t}(a_t^2+1)^{2d_t} \cdot r_t^{-2d_t}(u_t - a_t)^{-4d_t} \bmod n_t \\
&= H_3(X)^{d_t}[(a_t u_t + 1)^2 + (u_t - a_t)^2]^{2d_t}(u_t - a_t)^{-4d_t} \bmod n_t \\
&= H_3(X)^{d_t}[(c^2+1)(u_t - a_t)^2]^{2d_t}(u_t - a_t)^{-4d_t} \bmod n_t \\
&= H_3(X)^{d_t}(c^2+1)^{2d_t}
\end{aligned}$$

The above reduction shows that the values $\{b_t, r_t, u_t\}$ that are learned from corresponding signature process record tuple $\{X, Y, a_t, \beta, t\}$ are eliminated. This shows that the $s = t.r_t^2.r_t'^4 \; mod \; n_t$ holds true for any values $\{b_t, r_t, u_t\}$ that are learned from other signature process records tuples. Therefore, the TPSP cannot link the token ID to a specific customer's signature process record using the unblinded signature. □

6.4 Unforgeability

Unforgeability in the proposed protocol ensures that the digital signatures that are issued to authorise payment tokens cannot be created or replicated by any malicious adversaries. We aim to demonstrate that adversaries can't forge the TPSP's signature using the following proof.

Theorem 6.4. *The proposed partially blind signature scheme is unforgeable under chosen message attacks, assuming the RSA assumption holds.*

Proof. We assume that there exists a PPT (Probabilistic Polynomial-Time) adversary who can break the unforgeability of the scheme with a non-negligible probability ε. We will construct a PPT algorithm AlgoUF that utilises this adversary to solve the RSA problem with non-negligible probability, thereby contradicting the RSA assumption. The adversary, in this case, can eavesdrop on the communication between the customer, merchant, and TPSP, but cannot modify or inject messages as per the proofs provided above. The adversary's goal is to forge a new valid signature (c', s') for a different token ID $T'id$ (with common information X') that passes the signature verification process.

The steps involved in the AlgoUF algorithm are as follows:

(1) AlgoUF receives an RSA challenge (N, e, y). It sets the public parameters of the signature scheme as $(e_t, n_t) = (e, N)$ and $H3$ as a random oracle.
(2) AlgoUF simulates the TPSP and interacts with the adversary as follows: When the adversary makes a signing query on (X_i, TID_i) AlgoUF chooses a random $(r_i, u_i \in Z_N^*)$, computes $Y_i = r_i^e H3(TID_i)(u_i^2 + 1)(mod\ N)$, and sends Y_i to the adversary. The adversary responds with β_i. AlgoUF then computes $t_i = (H3(X_i)^d (Y_i(a_i^2 + 1)\beta_i^{-2}))^{2d}(mod\ N)$, where d is the RSA private key (which AlgoUF does not know). Finally, AlgoUF sends (β_i^{-1}, t_i) to the adversary. Note that this simulation is perfect since AlgoUF can compute all the values that the TPSP would compute without knowing the private key.
(3) The adversary eventually will output a forgery (X^*, TID^*, c^*, s^*) such that TID^* is different from all TID_i that are sent for query to the signing oracle and the signature verification equation holds:
(4) AlgoUF now uses the forgery to compute the e-th root of y as follows.
- Case 1: X^* is new
  - AlgoUF programs the random oracle $H3$ such that $H3(X^*) = y$. From the forgery, AlgoUF has $(s^*)^e \equiv yH_3(TID^*)^2((c^*)^2+1)^2 \pmod N$. AlgoUF then can compute $H_3(TID^*)^2((c^*)^2+1)^2 \pmod N$ since it knows all the values involved. Let $z \equiv H_3(TID^*)^2((c^*)^2 + 1)^2 \pmod N$. Then, $(s^*)^e \equiv yz \pmod N$. AlgoUF can compute $z^{-1} \pmod N$ using the Extended Euclidean Algorithm. Finally AlgoUF computes $x = s^* z^{-1} \pmod N$. This x satisfies $x^e \equiv y \pmod N$ thus solving the RSA challenge.
- Case 2: X^* is not new
  - In this case the adversary must have made a signing query on X^* before. Let the corresponding values be (X^*, TID_i, c_i, s_i). From the forgery and the previous signature, AlgoUF has two equations, a) $(s^*)^e \equiv H_3(X^*)H_3(TID^*)^2((c^*)^2 + 1)^2 \pmod N$ and b) $s_i^e \equiv H_3(X^*)H_3(TID_i)^2(c_i^2 + 1)^2 \pmod N$. If we divide this we get,

$$(\frac{s_i}{s^*})^e \equiv \frac{H_3(TID_i)^2(c_i^2 + 1)^2}{H_3(TID^*)^2((c^*)^2 + 1)^2} \pmod N$$

    AlgoUF knows all the values on the right side of the equation and can compute that value $z = (\frac{s_i}{s^*})^e \equiv z \pmod N$. AlgoUF can compute the $s_i^{-1} \pmod N$ and $z^{-1} \pmod N$ using the Extended Euclidean Algorithm. Finally AlgoUF computes $x = \frac{s_i}{s^*} z^{-1} \pmod N$. This x satisfies $x^e \equiv 1 \pmod N$. Now, AlgoUF can pick any previous query (X_j, TID_j, c_j, s_j) where X_j is different from X^*. It can similarly divide the verification equations for the forgery and this query to get another equation of the form

$$(\frac{s_j}{s^*})^e \equiv z' \pmod N$$

    and compute $x' = \frac{s_j}{s^*} z'^{-1} \pmod N$. This x' will satisfy $x'^e \equiv \frac{H_3(X_j)}{H_3(X^*)} \pmod N$. Since AlgoUF knows the random oracle $H3$, it also knows the discrete log of $\frac{H_3(X_j)}{H_3(X^*)}$ with respect to the base 'g' (the generator of the group used in the random oracle). If $\frac{H_3(X_j)}{H_3(X^*)} = g^a$, then,$x'^e \equiv g^a \pmod N$. Finally, AlgoUF computes the e-th root of y as $x'^{-a} \cdot y \pmod N$.

$$(s^*)^e \equiv H_3(X^*)H_3(TID^*)^2((c^*)^2 + 1)^2 \pmod N$$

If the adversary succeeds in forging a signature with probability ϵ, then the algorithm AlgoUF succeeds in solving the RSA challenge with probability at least $\epsilon/2$ (considering both cases). This contradicts the assumed hardness of the RSA problem which proposes that factoring large

RSA moduli is computationally infeasible. Hence, we conclude that no adversary can forge valid signatures within our payment system. □

6.5 Exculpability

The exculpability property of our protocol ensures that the merchant/customer cannot fraudulently reuse (double-spend) the payment token. It also allows the TPSP to utilise the security tag to reveal the identity of the customer attempting to double-spend.

6.5.1 Merchant-Double Spending. Our protocol employs ZKP-based mutual authentication, removing the need for TPSPs during authentication between the customer and the merchant. Before using a payment token with a new merchant, the customer must prove that they are a registered user in the system. This is achieved through a ZKP-based challenge-response exchange, where the merchant generates a unique challenge for each transaction, and the customer computes a response using their secret keys. The same response cannot be reused for other transactions. Thus, even if a merchant tries to impersonate the customer and reuse the token, they'll fail because they can't generate the correct challenge response. Hence, in our proposed untraceable payment scheme merchants cannot engage in double-spending activities.

6.5.2 Customer-Double Spending. Consider a scenario where a customer attempts to double-spend a payment token, trying to spend it concurrently with two different merchants. In our untraceable payment scheme, only one merchant's deposit request will be successful, while the other will be denied. While both merchants are not financially disadvantaged, this scenario allows the dishonest customer to remain anonymous. To address this limitation and mitigate potential abuse, our protocol incorporates a tracing mechanism that enables the identification of such dishonest customers, even within the untraceable payment framework. To implement this mechanism, the customer generates a security tag, denoted as T_c, using R and their public key, P_{key}, as follows:

$$T_c = P_{key}^{(1+R)}$$

If a Token ID, denoted as T_{ID}, already exists within the TPSP's database, the anonymity revocation protocol is initiated. For instance, if a customer submits the same Token ID to merchants A and B, the respective merchants will store (T_{ID}, T_1, R_1) and (T_{ID}, T_2, R_2), where T_1 and T_2 are security tags created using random numbers R_1 and R_2, respectively. The TPSP can then request the merchants who submitted the same Token ID to provide the security tags and corresponding random numbers. This allows the TPSP to reveal the identity of the customer by calculating:

$$\begin{aligned}
&\left(\frac{T_2^{R_1}}{T_1^{R_2}}\right)^{(R_1-R_2)^{-1}} \\
&= \left(\frac{(P_{key}^{(1+R_2)})^{R_1}}{(P_{key}^{(1+R_1)})^{R_2}}\right)^{(R_1-R_2)^{-1}} \\
&= \left(P_{key}^{(R_1+R_1R_2-R_2-R_2R_1)}\right)^{(R_1-R_2)^{-1}} \\
&= \left(P_{key}^{(R_1-R_2)}\right)^{(R_1-R_2)^{-1}} \\
&= P_{key}
\end{aligned}$$

where P_{key} is the public key of the dishonest customer.

7 PROTOCOL MODELLING IN PROVERIF

In this section, we model the protocol's security and privacy properties using ProVerif. Security and privacy properties in protocols are typically categorised into trace properties and equivalence properties in Proverif. Trace properties must hold for every execution of the protocol. Examples of trace properties include secrecy and authentication. On the other hand, equivalence properties are evaluated by comparing concurrent executions of the protocol. These properties ensure that an attacker cannot distinguish between similar executions. Untraceability falls under the category of equivalence properties. In our ProVerif model, we treat cryptographic primitives like hashing, signatures, encryption and decryption as perfectly secure black boxes.

7.1 Formalising The Protocol

In this section, we present the symbolic formal model of our untraceable payment scheme, which involves three actors: the customer (C), the merchant (M), and the TPSP (S). We assume a Dolev-Yao [10] attacker who can eavesdrop, modify, inject, or remove messages shared on public channels but only if it possesses the correct key. Thus, the protocol assumes perfect cryptography and restricts the attacker's capabilities. Additionally, we distinguish between honest and fraudulent actors in our payment scheme. Honest actors will follow the protocol and fulfil their tasks, while fraudulent actors will perform malicious actions such as double-spending and disregarding the protocol.

To model communication between the customer, merchant, and TPSP, we introduce a public channel *c*. The channel *c* represents the real-world communication medium. Messages transmitted over the public channel can be encrypted and decrypted by the protocol participants. We represent cryptographic primitives in our model as symbolic functions. Specifically, we use a constructor *senc* to represent symmetric encryption and a destructor *sdec* to represent symmetric decryption. We model digital signatures using a well-known ProVerif pattern in combination with our functional model. Digital signatures, like asymmetric encryption, rely on private (*sskey*) and public key (*spkey*). To denote the generation of key pairs, we use a constructor *spk* which takes *sskey* and returns *spkey*. Signing a digital signature is represented by a constructor *sign*, which takes a *bitstring* and a *sskey* as inputs and returns a *bitstring* signature as output. We use a destructor *checksign* to verify the signature, which returns *true* if the verification is successful. The *blind* and *unblind* functions represent blinding a message and unblinding a blinded message to obtain the original message. We model our protocol by representing each participant (customer, merchant, TPSP) as a separate process. We assume the authentication phase has been completed and that participants have securely exchanged the necessary symmetric keys for encrypted communication.

7.1.1 Customer Process. Initially, the customer sends an encrypted Purchase Request (*PR*) to the merchant. After receiving the Purchase Request-Reply (*PRR*), the customer generates a Token ID (*TD*) and blinds it. The encrypted blinded Token ID is then sent to the TPSP, and the customer waits for the TPSP to reply with a blind signature. After receiving the blind signature, the customer sends an encrypted payment token along with the unblinded signature to the merchant. To indicate the start of each phase of the protocol, we introduce events such as *PurchaseRequest*, *TokenWithdrawal* and *PaymentbyToken*.

The customer process can be formalised as follows:

```
let CUSTOMER(k1:key, k2:key, r:bitstring)=
event PurchaseRequest;
out(c, senc(PR,k1));
in (c, y:bitstring);
let PRR= sdec(y, k1) in
event TokenWithdrawal;
out(c, senc(blind (TD, r), k2));
in(c, x:bitstring);
let BS=sdec(x, k2) in
let sign=unblind(BS, r) in
event PaymentbyToken;
out(c, (senc(TD, k1), senc(sign, k1)));
```

7.1.2 Merchant Process. The merchant first receives the Purchase Request (*PR*) from the customer and sends a Purchase Request-Reply tuple *PRR* back to the customer. After receiving the payment token from the customer, the merchant verifies the signature on the token. If the verification is successful, the merchant sends a deposit request to the TPSP and waits for a reply. After receiving the Deposit Confirmation *DC* message from the TPSP, the merchant sends the Purchase Confirmation *PC* message to the customer. To indicate the commencement of each phase of the protocol, we insert events such as *PurchaseRequest*, *PaymentbyToken* and *DepositRequest*. The formalisation of the merchant process is given below:

```
let MERCHANT(k1:key, k3:key, pkS:spkey)=
event PurchaseRequest;
in (c, y:bitstring)
let PR = sdec(y,k1) in
out (c, senc(PRR,k1);
event PaymentbyToken;
in (c, (x:bitstring, z:bitstring));
let TD=sdec(x, k1) in
let sign=sdec(z, k1) in
let (=pkS, k:key) = checksign(sign, pkS) in
event DepositRequest;
out(c, (senc(TD, k3),senc(sign, k3)));
in(c, w:bitstring);
let DC = sdec(w, k3) in
out(c, senc(PC, k1) ;
```

7.1.3 TPSP. The TPSP receives the encrypted blinded Token ID from the customer and issues a blind signature on it. It sends the encrypted signature back to the customer. Upon receiving the encrypted signature, the customer unblinds it to obtain the actual signature on the payment token. The customer then sends the payment token and unblinded signature to the merchant, who in turn sends it to the TPSP to initiate the deposit process. We insert events such as *TokenWithdrawal* and *DepositRequest* to indicate the start of each phase of the protocol. The TPSP process can be formalised as follows:

```
let TPSP(k2:key, k3:key, skS:sskey)=
event TokenWithdrawal;
in (c, y:bitstring)
let TD = sdec(y, k2) in
out(c, senc(sign(TD, skS), k2));
event DepositRequest;
in(c, (x:bitstring, z:bitstring));
let TD = sdec(x, k3) in
let sign =sdec(z, k3) in
out(c, senc(DC, k3) ;
```

7.2 Model Evaluation

We verify three important properties such as confidentiality, message authenticity and untraceability using proverif.

7.2.1 Confidentiality. We use ProVerif's reachability property to verify message confidentiality. We model the property with a predicate query: $query\ attacker(s)$, where "s" represents the message in question. This query essentially determines if the attacker can access any messages exchanged by the protocol participants, indicating a confidentiality breach. ProVerif is used to assess the secrecy of all messages exchanged over the public channel in our protocol, both sent and received by the actors. The secrecy evaluation results are presented in Table 2, where a (✓) indicates that the protocol upholds the secrecy property and a (×) indicates that the protocol does not uphold the property.

Table 2. Analysis of Secrecy Property

Message	Meaning	From	To	Result
PR	Purchase Request	Customer	Merchant	✓
PRR	Purchase Request Reply	Merchant	Customer	✓
TD	Token ID	Customer	TPSP	✓
BS	Blind Signature	TPSP	Customer	✓
DC	Deposit Confirmation	TPSP	Merchant	✓
PC	Purchase Confirmation	Merchant	Customer	✓

7.2.2 Message Authenticity. We verify message authenticity using ProVerif's correspondence assertion property. Correspondence assertions formalise the relationship between events in a protocol execution. For example, the correspondence property can specify that if event e occurs, it implies the prior occurrence of event e'. In our protocol, we declare the following events:

- *PurchaseRequest* - The customer has authenticated themselves successfully with the merchant to commence the purchase.
- *TokenWithdrawal* - The customer can initiate the token withdrawal process with the TPSP after receiving a purchase reply from the merchant. The customer is authorised to perform this event only if it has been authenticated by both the merchant and the TPSP. Therefore, the execution of the *PurchaseRequest* event must precede the execution of the *TokenWithdrawal* event.
- *PaymentbyToken* - The occurrence of this event indicates that the TPSP has successfully issued a valid signature on the token and that the customer can proceed with payment. To

carry out this event, the customer must have obtained a valid payment token. Therefore, the execution of the *TokenWithdrawal* event must precede the execution of the *PaymentbyToken* event.

- *DepositRequest* - The occurrence of this event signifies that the merchant has verified the payment token and is ready to deposit it with the TPSP. For a merchant to execute this event, they must have authenticated themselves both with the TPSP and the customer. Therefore, the *PaymentbyToken* event must precede the *DepositRequest* event.

From Table 3 we can observe that the authenticity property of the protocol is upheld during each stages.

Table 3. Analysis of Authenticity Property

Event Correspondence	Result
PurchaseRequest-TokenWithdrawal	✓
TokenWithdrawal-PaymentbyToken	✓
PaymentbyToken-DepositRequest	✓

7.2.3 Untraceability. Our protocol adheres to the established definition of untraceability, ensuring that the TPSP cannot link a customer's identity to a specific Token ID. We formally evaluate this property within our ProVerif model.

Definition 7.1. For any two digital tokens $T1$ and $T2$ and for two customer processes $C1$ and $C2$ such that $T1$ and $T2$ are generated by $C1$ and $C2$ respectively, the untraceability property is upheld in our protocol if the TPSP cannot distinguish whether tokens T_i where $i = 1$ *or* 2 is generated by $C1$ or $C2$.

We demonstrate untraceability using the concept of observational equivalence. Specifically, our protocol satisfies the untraceability requirement if and only if the customer process $C1$ with Token ID $T1$ executed in parallel with another customer process $C2$ with Token ID $T2$ is observationally equivalent to the customer process $C1$ with Token ID $T2$ executed in parallel with customer process $C2$ with Token ID $T1$. In other words, we can express this as: $C1 \mid C2 \approx C1T2/T1 \mid C2T1/T2$. ProVerif confirms that this untraceability property is upheld during each execution of our protocol.

8 EVALUATION

8.1 Computational Complexity Analysis

This section presents the comparative analysis of the computational performance of selected schemes [8, 26, 28] against the scheme we proposed. We limit our performance comparison to these payment systems because they share similar stages with our proposed scheme. Our proposed scheme consists of different stages such as registration, token withdrawal, payment and token deposit. These stages involve various cryptographic operations such as issuing and verifying digital signatures, encryption, decryption and relevant key generation. In Table 4 we present the number of significant operations such as exponents (E), hashes (H), modular multiplication (M) required for the proposed scheme and other similar schemes [8, 26, 28]. When comparing we consider the customer, merchant and TPSP as one single entity. We also provide another comparison in Table 5 in terms of computational cost where M, H, E etc. are in a 1024-bit modulus. For the computational cost estimation, we follow the process provided by [32] and assume that $E \approx 240M$ and $H \approx M$.

Table 4. Complexity comparison based on number of operations

Protocol Stage	[26]	[28]	[8]	Proposed Scheme
Registration	$2E + 2M + 2H$	$18E + 16M$	$2E + 3M$	$2E + 2M + 2H$
Token Withdrawal	$7E + 9M$	$10E + 9M$	$8E + 8M + 3H$	$6E + 6M + 1H$
Payment	$5E + 5M + 1H$	$8E + 8M$	$5E + 5M + 1H$	$2E + 1M + 2H$
Token Deposit	$1E + 2M$	$5E + 5M$	$4E + 2M + 3H$	$3E + 1M + 1H$

Table 5. Complexity comparison based on computational cost

Protocol Stage	[26]	[28]	[8]	Proposed Scheme
Registration	$\approx 484M$	$\approx 4336M$	$\approx 483M$	≈ 484
Token Withdrawal	$\approx 1689M$	$\approx 2409M$	$\approx 1931M$	≈ 1447
Payment	$\approx 1206M$	$\approx 1928M$	$\approx 1206M$	≈ 483
Token Deposit	$\approx 242M$	$\approx 1205M$	$\approx 965M$	≈ 722
Total Cost	$3137M$	$5542M$	$4102M$	$3136M$

The performance analysis presented in Tables 4 and 5 underscores the efficiency of our proposed untraceable payment scheme. The results show that our proposed scheme causes the lowest overall computational cost when compared to existing schemes [8, 26, 28]. The design choices we have made in this scheme such as partially blind signatures and a lightweight zero-knowledge proof mechanism contribute to the computational cost being lower than other schemes. Low computational cost enhances our scheme's practicality and suitability for real-world mobile payment scenarios, where resource constraints are often a concern.

8.2 Security and Privacy

Table 6 presents a comparative analysis of online [7, 29, 42] and offline [4, 8, 26, 39] payment systems with our proposed scheme and demonstrate its alignment with the evaluation criteria.

Payment scheme	C1	C2-a	C2-b	C3	C4	C5	C6	C7
Online schemes								
Eszczyna et al. [29]	×	×	×	×	✓	✓	×	×
Cao et al. [7]	×	✓	✓	✓	✓	✓	✓	×
Tso [42]	×	×	✓	✓	✓	✓	×	×
Offline schemes								
Kutubi et al. [26]	×	✓	✓	✓	✓	✓	✓	×
Chang et al. [8]	×	×	✓	✓	✓	✓	×	×
Kutubi et al. [28]	×	✓	✓	✓	✓	✓	×	×
Our proposed scheme	✓	✓	✓	✓	✓	✓	✓	✓

Table 6. Comparison of Untraceable Payment Schemes: C1-Complete Untraceability, C2-a-Exculpability-Merchant Double Spending, C2-b-Exculpability-Customer Double Spending, C3-Unforgeability, C4-Confidentiality, C5-Message Authenticity, C6-Efficiency, C7-Regulatory Compliance

In Table 6 a (✓) indicates that the protocol upholds the specific criteria and a (×) indicates that the protocol does not uphold that criteria. This comparative analysis highlights the strengths of our proposed scheme in addressing a wider array of security and privacy concerns compared to the existing solutions. Online payment systems often operate under the assumption that the TPSP or Bank is inherently trustworthy. They mainly focus on safeguarding user privacy against a curious merchant or external adversaries. In contrast, offline payment systems do not make this trust assumption regarding the TPSP. Instead, they introduce additional third parties, such as certificate authorities or anonymous identity providers, to protect user privacy. However, these systems inherently assume that these third parties will not collude with a curious TPSP to share user-related information. Hence, as Table 6 shows, the majority of these systems fall short of meeting the untraceability criterion when it comes to a curious TPSP.

Additionally, some works [8, 29, 42] does not explicitly state whether their proposed schemes offer protection against merchant double-spending. Similarly, [29] does not explicitly address whether their scheme protects against customer double-spending. Furthermore, [29] also fails to specify whether their scheme ensures that only authorised parties can create payment tokens. Some works [29, 42] did not provide a detailed description of their computational cost to demonstrate the efficiency of their systems. At the same time, Section 8.1 showcases that our scheme outperforms [8, 28] in terms of computational efficiency while maintaining a similar computational complexity to [26]. Regarding regulatory compliance, all the works fall short of providing explicit mechanisms to address this concern. While they prioritise user anonymity, they do not adequately address the critical aspect of verifying user identities or managing illicit activities like money laundering. This lack of clarity raises concerns about their practicality in real-world scenarios where regulatory bodies mandate certain checks and balances to prevent financial crimes. In our proposed scheme we aim to strike a balance between user privacy and regulatory compliance by allowing the TPSP to collect basic information required to register a user and the corresponding transaction amount.

However, it's crucial to acknowledge the limitations discussed in Section 5.1 when it comes to our proposed scheme. Our proposed scheme does not address adversaries who might compromise user devices to gain access to secret keys. While the zero-knowledge proof mechanism offers some protection against a compromised merchant device, it doesn't prevent fraudulent transactions if a customer's device is compromised. This limitation highlights the persistent challenge of safeguarding user privacy and system security in the face of evolving threats. Therefore, in our future work, where we plan to implement a working solution for real-world scenarios, we intend to address this limitation by utilising secure hardware elements such as a Trusted Execution Environment (TEE) or a Secure Element (SE).

9 CONCLUSION

This paper introduces a novel mobile payment scheme that prioritises user privacy while accommodating necessary regulatory oversight. The scheme employs cryptographic primitives such as blind signatures and zero-knowledge proofs to limit the Third-Party Service Provider's (TPSP) access to sensitive transaction related data We ensured that the TPSP cannot track user spending patterns or link transactions to individual customers. Our proposed scheme also includes mechanisms to prevent double-spending by both customers and merchants, further enhancing its security against fraudulent activities. The paper also introduces a comprehensive evaluation framework consisting of key criteria: complete untraceability, exculpability-merchant double-spending, exculpability customer double-spending, unforgeability, confidentiality, message authenticity, efficiency and regulatory compliance. This framework helps to rigorously assess the security and privacy characteristics of centralised untraceable payment systems, offering a structured methodology to analyse and compare different schemes, promoting transparency. Complexity based formal analysis and

ProVerif based modelling further validates the scheme's robustness in upholding its privacy and security goals. In the future work we are fully committed to pursuing the practical evaluation of our system. The prototype we are planning to develop will be tested in various simulated transaction scenarios to evaluate it's performance under different conditions.

REFERENCES

[1] Masayuki Abe and Tatsuaki Okamoto. 2000. Provably secure partially blind signatures. In *Advances in Cryptology—CRYPTO 2000: 20th Annual International Cryptology Conference Santa Barbara, California, USA, August 20–24, 2000 Proceedings 20*. Springer, 271–286.

[2] Kate Allman. 2018. The dark side of Bitcoin. *LSJ: Law Society Journal* 42 (2018), 28–29.

[3] Man Ho Au, Willy Susilo, and Yi Mu. 2007. Practical compact e-cash. In *Information Security and Privacy: 12th Australasian Conference, ACISP 2007, Townsville, Australia, July 2-4, 2007. Proceedings 12*. Springer, 431–445.

[4] Yaser Baseri, Benyamin Takhtaei, and Javad Mohajeri. 2013. Secure untraceable off-line electronic cash system. *Scientia Iranica* 20, 3 (2013), 637–646.

[5] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. 2005. Compact e-cash. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 302–321.

[6] Jan Camenisch and Anna Lysyanskaya. 2004. Signature schemes and anonymous credentials from bilinear maps. In *Annual international cryptology conference*. Springer, 56–72.

[7] Chenglong Cao and Xiaoling Zhu. 2019. Strong anonymous mobile payment against curious third-party provider. *Electronic commerce research* 19 (2019), 501–520.

[8] Ya-Fen Chang, Wei-Liang Tai, Yao-Ching Liu, and Huan-Wen Chen. 2018. Vulnerability of Baseri et al.'s Untraceable Offline Electronic Cash System. In *2018 International Conference on System Science and Engineering (ICSSE)*. IEEE, 1–5.

[9] Hung-Yu Chien, Jinn-Ke Jan, and Yuh-Min Tseng. 2001. RSA-based partially blind signature with low computation. In *Proceedings. Eighth International Conference on Parallel and Distributed Systems. ICPADS 2001*. IEEE, 385–389.

[10] Danny Dolev and Andrew Yao. 1983. On the security of public key protocols. *IEEE Transactions on information theory* 29, 2 (1983), 198–208.

[11] David Easley, Maureen O'Hara, and Soumya Basu. 2019. From mining to markets: The evolution of bitcoin transaction fees. *Journal of Financial Economics* 134, 1 (2019), 91–109.

[12] Sean Foley, Jonathan R Karlsen, and Tālis J Putniņš. 2019. Sex, drugs, and bitcoin: How much illegal activity is financed through cryptocurrencies? *The Review of Financial Studies* 32, 5 (2019), 1798–1853.

[13] Zhonghui Ge, Jiayuan Gu, Chenke Wang, Yu Long, Xian Xu, and Dawu Gu. 2023. Accio: Variable-Amount, Optimized-Unlinkable and NIZK-Free Off-Chain Payments via Hubs. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 1541–1555.

[14] Shafi Goldwasser, Silvio Micali, and Chales Rackoff. 2019. *The knowledge complexity of interactive proof-systems*. ACM. 203–225 pages.

[15] Yoshikazu Hanatani, Yuichi Komano, Kazuo Ohta, and Noboru Kunihiro. 2007. Provably secure untraceable electronic cash against insider attacks. *IEICE transactions on fundamentals of electronics, communications and computer sciences* 90, 5 (2007), 980–991.

[16] Zahra Hatefi, Majid Bayat, Mahdi R Alaghband, Negin Hamian, and Seyed Morteza Pournaghi. 2023. A conditional privacy-preserving fair electronic payment scheme based on blockchain without trusted third party. *Journal of Ambient Intelligence and Humanized Computing* 14, 8 (2023), 10089–10102.

[17] Ethan Heilman, Alison Kendler, Aviv Zohar, and Sharon Goldberg. 2015. Eclipse attacks on bitcoin's peer-to-peer network. In *24th {USENIX} Security Symposium ({USENIX} Security 15)*. 129–144.

[18] Xiaoming Hu and Shangteng Huang. 2008. Analysis of ID-based restrictive partially blind signatures and applications. *Journal of Systems and Software* 81, 11 (2008), 1951–1954.

[19] SK Hafizul Islam, Ruhul Amin, GP Biswas, Mohammad S Obaidat, and Muhammad Khurram Khan. 2016. Provably secure pairing-free identity-based partially blind signature scheme and its application in online e-cash system. *Arabian Journal for Science and Engineering* 41 (2016), 3163–3176.

[20] Ravi Shankar Jha, Saleh Umar, Tokpe Kossi, and Ouattara Mohamad Lamine. 2023. Token Bases Valid and Secure Payment System Using SHA-256. In *Advancements in Interdisciplinary Research: First International Conference, AIR 2022, Prayagraj, India, May 6–7, 2022, Revised Selected Papers*. Springer, 29–38.

[21] Ghassan O Karame, Elli Androulaki, and Srdjan Capkun. 2012. Double-spending fast payments in bitcoin. In *Proceedings of the 2012 ACM conference on Computer and communications security*. 906–917.

[22] Heikki Karjaluoto, Aijaz A Shaikh, Matti Leppäniemi, and Roope Luomala. 2020. Examining consumers' usage intention of contactless payment systems. *International Journal of Bank Marketing* 38, 2 (2020), 332–351.

[23] Frode Kjærland, Aras Khazal, Erlend A Krogstad, Frans BG Nordstrøm, and Are Oust. 2018. An analysis of bitcoin's price dynamics. *Journal of Risk and Financial Management* 11, 4 (2018), 63.
[24] Nir Kshetri and Joanna F DeFranco. 2020. Is Privacy Dead? *IT Professional* 22, 5 (2020), 4–12.
[25] Mahender Kumar and CP Katti. 2016. An efficient ID-based partially blind signature scheme and application in electronic-cash payment system. *Accent. Trans. Inf. Secur* 2, 6 (2016), 36–42.
[26] Md Abdullah Al Rahat Kutubi, Kazi Md Rokibul Alam, and Yasuhiko Morimoto. 2021. A simplified scheme for secure offline electronic payment systems. *High-Confidence Computing* 1, 2 (2021), 100031.
[27] Md. Abdullah Al Rahat Kutubi, Kazi Md. Rokibul Alam, and Yasuhiko Morimoto. 2021. A simplified scheme for secure offline electronic payment systems. *High-Confidence Computing* 1, 2 (2021), 100031. https://doi.org/10.1016/j.hcc.2021.100031
[28] Md Abdullah Al Rahat Kutubi, Kazi Md Rokibul Alam, Rafaf Tahsin, GG Ali, Peter Han Joo Chong, and Yasuhiko Morimoto. 2017. An offline electronic payment system based on an untraceable blind signature scheme. *KSII Transactions on Internet and Information Systems (TIIS)* 11, 5 (2017), 2628–2645.
[29] Rafał Leszczyna. 2024. Activity-based payments: alternative (anonymous) online payment model. *International Journal of Information Security* (2024), 1–19.
[30] Chao Lin, Debiao He, Xinyi Huang, Muhammad Khurram Khan, and Kim-Kwang Raymond Choo. 2020. DCAP: A secure and efficient decentralized conditional anonymous payment system based on blockchain. *IEEE Transactions on Information Forensics and Security* 15 (2020), 2440–2452.
[31] Limin Ma, Yu Ge, and Yuesheng Zhu. 2014. TinyZKP: a lightweight authentication scheme based on zero-knowledge proof for wireless body area networks. *Wireless personal communications* 77 (2014), 1077–1090.
[32] Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. 2018. *Handbook of applied cryptography*. CRC press.
[33] Ian Miers, Christina Garman, Matthew Green, and Aviel D Rubin. 2013. Zerocoin: Anonymous distributed e-cash from bitcoin. In *2013 IEEE Symposium on Security and Privacy*. IEEE, 397–411.
[34] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. *Decentralized business review* (2008), 21260.
[35] Martin Pirker and Daniel Slamanig. 2012. A Framework for Privacy-Preserving Mobile Payment on Security Enhanced ARM TrustZone Platforms. In *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*. 1155–1160. https://doi.org/10.1109/TrustCom.2012.28
[36] Sören Preibusch, Thomas Peetz, Gunes Acar, and Bettina Berendt. 2016. Shopping for privacy: Purchase details leaked to PayPal. *Electronic Commerce Research and Applications* 15 (2016), 52–64.
[37] Untung Rahardja, Qurotul Aini, Eka Purnama Harahap, and Raihan Raihan. 2021. GOOD, bad and dark bitcoin: a systematic literature review. *Aptisi Transactions on Technopreneurship (ATT)* 3, 2 (2021), 115–119.
[38] Balaji Rajendran, Anoop Kumar Pandey, and BS Bindhumadhava. 2017. Secure and privacy preserving digital payment. In *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. IEEE, 1–5.
[39] Aye Mi San and Chanboon Sathitwiriyawong. 2016. Privacy-preserving offline mobile payment protocol based on NFC. In *2016 International Computer Science and Engineering Conference (ICSEC)*. IEEE, 1–5.
[40] Statista. 2023. Mobile commerce retail sales worldwide from 2021 to 2026. https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales-mobile/
[41] Tidio. 2023. Mobile Commerce Statistics [2023]: Trends, Growth, Predictions. https://www.tidio.com/blog/mobile-commerce-statistics/
[42] Raylin Tso. 2018. Untraceable and anonymous mobile payment scheme based on near field communication. *Symmetry* 10, 12 (2018), 685.
[43] Ding Wang and Ping Wang. 2016. Two birds with one stone: Two-factor authentication with security beyond conventional bound. *IEEE transactions on dependable and secure computing* 15, 4 (2016), 708–722.
[44] Hua Wang and Yanchuan Zhang. 2001. Untraceable off-line electronic cash flow in e-commerce. In *Proceedings 24th Australian Computer Science Conference. ACSC 2001*. IEEE, 191–198.
[45] Fangguo Zhang and Kwangjo Kim. 2003. Efficient ID-based blind signature and proxy signature from bilinear pairings. In *Information Security and Privacy: 8th Australasian Conference, ACISP 2003 Wollongong, Australia, July 9–11, 2003 Proceedings 8*. Springer, 312–323.
[46] Lin Zhong, Qianhong Wu, Jan Xie, Jin Li, and Bo Qin. 2019. A secure versatile light payment system based on blockchain. *Future Generation Computer Systems* 93 (2019), 327–337.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009