Securing Privacy in Offline Payment for Retail Central Bank Digital Currency: A Comprehensive Framework

Olivier Atangana, Morgan Barbier, Lyes Khoukhi, Willy Royer

(017)

Securing Privacy in Offline Payment for Retail Central Bank Digital Currency: A Comprehensive Framework

O. Atangana [1,2], M. Barbier [1], L. Khoukhi [1] and W. Royer [2]
[1] University of Caen, ENSICAEN School Engineering, GREYC Laboratory,
6 Boulevard Maréchal Juin, 14050 Caen Cedex 4, France
[2] FIME EMEA, 9 Rue Commodore J H Hallet, 14000, Caen, France
Tel.: (+33)2 50 85 75 00
E-mail: olivier.atangana@fime.com

Summary: Privacy is a key feature in the successful adoption of Central Bank Digital Currency (CBDC). However, ensuring privacy in this disruptive innovation, particularly in offline payments, presents several challenges. Due to the nature of offline transactions, where one or both parties may not be immediately linked to a central network, security must be guaranteed with the utmost reliability. This study focuses on the development of an innovative protocol for Central Bank Digital Currencies, ensuring privacy and security for offline payments. The foundation of this protocol makes use of blind signature technology, a method that keeps a message's content secret from the signer and upholds the privacy of transaction Data. The zk-SNARK (Zero-knowledge Succint Non-interactive Argument of Knowledge) protocol, which assures that transactions are both private and verifiable without necessitating interaction between the prover and verifier, provides a complement to this. By leveraging blind signature technology and the zk-SNARK protocol, we explore how to overcome privacy-related challenges in retail CBDC while ensuring resilience against quantum attacks.

Keywords: CBDC, Privacy, Security, Blind signature, zk-SNARK, Offline payment function, Cash-like experience.

1. Introduction

1.1. Offline Function and Privacy in the Age of CBDC

In the wake of the 2008 financial crisis and the ongoing COVID-19 pandemic, the world experienced profound financial shifts. These events contributed to the emergence of cryptocurrencies, including Bitcoin, and accelerated the shift towards digital economies. In this context, the central bank digital currency (CBDC) has been identified as a reliable and credible form of digital money. However, while the use of CBDCs offers great promises, its implementation is facing a plurality of challenges that are intertwined with each other. Among others, offline payments and privacy, which has the potential to catalyze or hinder the adoption of CBDCs.

According to a survey conducted by the European Central Bank, privacy is the most desired attribute of a digital euro among both citizens and professionals [1]. This implies to face a plenty of concerns such as risk of profiling, citizen surveillance and dataveillance which tracks metadata by the implication of Big Data. Furthermore, offline payments are a popular characteristic among the general population, particularly among underbanked layers. This is mainly because it offers the possibility of improving financial inclusion, access, resilience, and maintaining a cash-like experience. Thus, reconciling privacy and offline payments is a real challenge.

To address this challenge, this paper presents a new framework using privacy-enhancing technologies without compromising offline transaction security while remaining resilient against quantum attacks. The remainder of this study is structured as follows. Section 2 presents the related work. In Section 3, we present a comparative study of related work and our proposal. We then present our offline privacy function approach in Section 4. Section 5 presents an analysis of the offline framework.

1.2. Main Contributions

Our research focuses on retail CBDC [2]. Notably, the token-based rCBDC which is designed to guard privacy. This offers a range of advantages, including end-to-end security and full privacy lifecycle protection. With this in mind, we aim to propose a new offline payment approach, CBDC, on a mobile device that combines a blind signature, the zk-SNARK protocol, and other strands of the literature, such as cryptography, open source technology, digital signature, public key infrastructure, privacy-by-design, and digital certificate. This approach guarantees security against double spending issues and preserves data privacy.

2. Related Work

Research on offline payment functions has attracted significant attention since the emergence of e-cash. This payment method is especially prominent in CBDC systems, offering advantages, such as resilience, cash-like characteristics, inclusion, universal access, and reduced transaction costs.

Despite this interest, few protocols or frameworks exist for offline CBDC payments in the current landscape.

An offline CBDC payment involves transferring value between devices without a ledger system connection, often in the absence of the Internet or telecom connectivity. The Offline Payment System (OPS) protocol proposed in [3] utilizes a two-tier certificate infrastructure established by a delegated financial authority. It employs cryptographic algorithms, enabling secure payments from Alice to Bob without Bob needing a security device. However, it may not fully meet the non-forgery and DDoS attack prevention requirements. An extension of the OPS using an ISO/IEC 7816-compatible smart card and NFC interface addresses these issues by setting transaction time limits and using cryptographic keys on the smart card [4]. Another proposal employs One-Time Programm (OTP) for each monetary unit, to ensure confidentiality, anonymity, and programmability [5]. However, vulnerabilities exist during OTP transmission and framework forgery risks. [6] proposes a flexible, programmable two-tier architecture solution that requires a secure TRE (Tamper Resistant Element) to counter hardware attacks, offers various transaction scenarios, and adaptable confidentiality levels. However, integrating the blockchain introduces de-anonymization risks. Another proposition suggests a universal, programmable solution resistant to quantum attacks, utilizing a new architecture, a trusted physical wallet and introducing "cyber coin" and "Mint" concepts [7]. It enables coin freezing in case of suspected criminal activity. An eID-based approach suggests that an electronic identity provider issue an e-wallet in a secure TEE (Trusted Execution Environment) chip [8]. Countersignatures divide transaction control between the payer and recipient, preventing the compromise of one device from affecting the entire transaction.

3. Materials and Method

3.1. Overview of Proposed Method

Our solution grounded in Free/Libre and Open Source Software (FLOSS), which enables source code modifications, vulnerability detection, and vendor independence. This fosters security, transparency, standardization, competition, and central bank transparency. A TEE hardware module complements the software aspect. Considering two clients, Alice and Bob, with trusted server-bank accounts, both methods use FLOSS-based trusted applications. Alice, requiring a secure device with TEE for fund transfers, benefits from data confidentiality, integrity, and atomicity. ARM TrustZone TEE technology was adopted [9]. Bob, a fund receiver, doesn't require a secure device. Secure and authenticated channels mediate all communications. Passwords and biometrics ensure access control, necessitating approval of spending. Secure storage thwarts the rollback. The proposed framework employs a two-tier system. The central bank manages the database, whereas the withdrawal and deposit protocols are relayed via commercial banks.

In our framework, CBDCs are encrypted amounts. Each coin corresponds to a public/private key pair, with the central bank's signature endowing it with a value. This signature is verified by the recipient's "public key." Offline transactions entail sending the signed coin alongside a unique zk-SNARK-based transaction proof, ensuring authenticity, invisibility, and confidentiality. This solution amalgamates FLOSS and TEE for secure and efficient CBDC transactions, encompassing user identification, offline capabilities, and transaction confidentiality. Table 1, shows a comparison table of these solutions with our approach.

3.2. Key Building Blocks

In the following section, we delve more deeply into the technology used in our framework. Our framework relies on the use of digital signature, blind signatures and zk-SNARK protocol.

Blind signature: According to Chaum, “a blind signature is used to create a cryptographic signature for a message without the signer learning the contents of the message being signed” [10]. This was used in our protocol for offline transactions. Leveraging user smartphone, each digital coin is 'blinded.’ Basically, “a user’s smartphone or other device can simply “blind” a desired number f(x) by multiplying it by a random number *b* that it chooses and raises to a denomination power” [11].

Zk-SNARK: To mitigate double-spending, the zk-SNARK protocol, a non-interactive variant of Zero-Knowledge Proof, validates the transmitted coins by guaranteeing the uniqueness of each transaction. In fact, "using a succinct argument, the client would be able to verify the correctness of the result, using resources that are significantly smaller than those necessary to perform the task from scratch" [12]. Subsequently, combined with the blind signature, the zk-SNARK protocol maintains confidentiality and ensures transaction legitimacy, adhering to the zk-SNARK principle used in Zcash [13].

Key exchange protocols: We employ the Diffie-Hellman key exchange protocol to establish a link between the coin and money returned after a transaction or refund.

Finally, two other technologies are used in our framework: digital signatures and digital certificates.

3.3. System’s Architecture

The system architecture requires a TEE and a certain number of functionalities. So, when Alice performs the onboarding step, the first registration step checks the TEE hardware and trusted OS for authenticity using a certificate from the equipment manufacturer (OEM). The initial functions of the system involve first setting up a device key pair,

resetting the balance and nonce, and confirming the key's origin. Then, it proceeds to check and save the server's certificate. After the main steps can commence. In Fig. 1, we outline the system with the various steps.

(1) is the withdraw. It consists of withdrawing of coins signed by the central bank and reduces Alice's online balance.

(2) is payment. This step performs offline encrypt transaction.

(3) is deposit. It involves depositing signed coins and incrementing the depositor's online balance.

(4) is exchange. Transactions in arrows 2 and 3 aren't one-offs. A client paid offline can use the coin for another offline payment without touching their online account.

All these steps will be explained in more detail in the next section.

Table 1. Comparison table.

CBDC Offline function	Wallet Type	No double Spending	Unforgeability	Aditionnal Hardware	Privacy Type	Risks and issues
cash	no	Yes	Yes	no	Data identity	Counterfeiting
[3]	Software	Yes	no	TEE	Data identity	-DDoS attack, Quantum attack, Physical attack -Limited transaction - Low privacy breach
[4]	Hardware	Yes	Yes	Trusted hardware module	Data identity	- Physical attack -Supply chain attack -Limited transaction - Low privacy breach
[5]	Hardware	Yes	Yes	TEE	Data privacy	-Physical attack -Framework could be hacked
[6]	Hardware & Software	Yes	Yes	TRE	Fully configurable level privacy	-Quantum attack -re-identification process in blockchain -Privacy breach depend on implementation
[7]	Hardware	Yes	Yes	Trusted Physical module	Data privacy	-Desintermediation of the banking system -Need to build a new architecture
[3]	Software	Yes	no	TEE	Data transaction	-Forged payment
Our approach	Free software	Yes	Yes	TEE	Data privacy	-TTP(bank) usurpation when the currency is signed

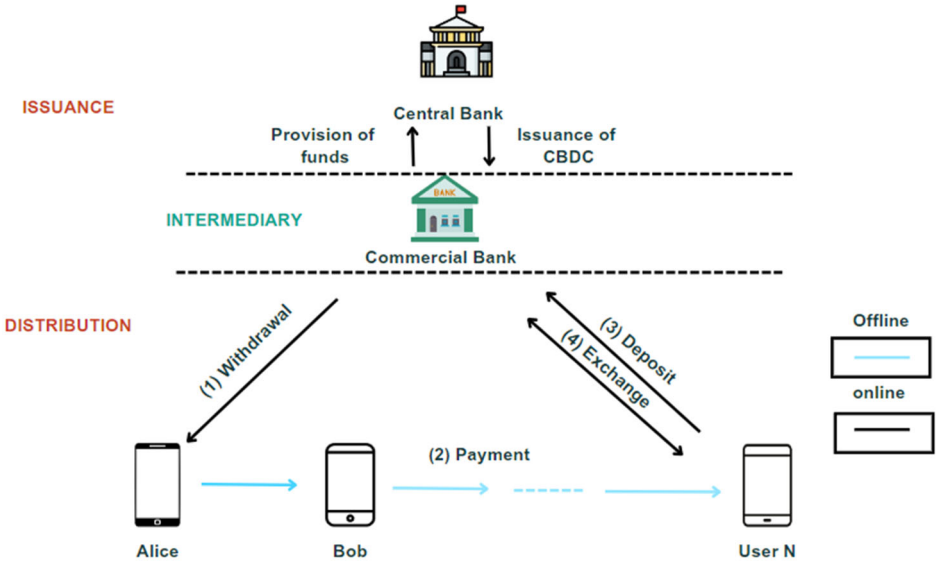

Fig. 1. Architecture of transferable electronic payment.

4. System Operation

Deploying the core functions of the system involves prior certification of the user as well as the application of the device he will be using. This is the wallet provisioning step. The remaining steps are related to the payment transaction. The main variables of the system are listed in Table 2.

Table 2. Notations in the proposed system.

Notations	Descriptions
(Vk_A, Sk_A)	Key pair for Alice
Vk_B	Bob's public key
(Jk, Zk)	zk-SNARK key pair
$cert_A$	Certificate generated by the server for vk_A
$cert_B$	Certificate generated by the server for Bob public's key
((e,n),d)	RSA keys which corresponds to specific coin values established by the central bank
b	Blinding factor
f	hash of the coin's public key (vk_A)
ff	Blinded hash of public key
s'	Blinded signature of ff
s	UnblindSignature
π	Proof string
$\vec{x}$	Public intput (as blinded signed coin)
$\vec{w}$	witness
db	Decision bit

4.1. Client Certification Protocol

The process begins with Alice generating a pair of signature keys (Vk, Sk) and sharing the authentication key with the server. The server then checks for a record associated with this key, and upon confirmation, the online balance is reset. Next, the server generates a certificate for Alice with server's private key, which incorporates the Alice's public key (Vk_A). This certificate is then sent to Alice. So, when she makes an offline payment using this method on their device, Bob can verify Alice's authenticity through Alice's certificate. This ensures a secure transaction can take place with confidence.

4.2. Secure Trusted Application (STA) Registration Protocol

Alice initiates the registration process by connecting her secure device to the server. She generates a key pair and verifies its authenticity with the server. If confirmed, the server creates a certificate for Alice's key. This certificate is both stored on the server and sent to Alice along with is then stored by the server and sent with a pair of a proving key JK and verification ZK to Alice. Upon obtaining the certificate, Alice proceeds to use her device to validate it. The device cross-verifies the certificate using the authentication key from the server, which was previously stored in the device. If the validation process is successful, the certificate is then securely stored on the device for future reference.

4.3. Withdraw Protocol

Algorithm 1. Coin's withdrawing.

Purpose: Alice intends to transfer x coins with blind signatures from her online balance, maintained on the S server, to her offline wallet.
Private inputs: b, d, Sk_A
Public inputs: f, ff, s', e, n, Vk_A
Output: s
1. Compute **f ← Hash(Vk_A)**
2. Calculate **ff ← BlindHashedKey(f, b, (e, n))**
3. Send **ff** to the server
4. Obtain **s' ← BlindlySignCoin(ff, d)**
5. Calculate **s ← UnblindSignature(s', b)**
6. Store the new minted coin with **Store_New_Coin(SK, s)**
7. Check if ff ≡ f^b e mod n
8. Verify the signature of the coin using the server's public key
9. If the signature is valid, proceed with the transaction
10. Update Alice's online balance and offline wallet accordingly

Algorithm 1 outlines the procedure by which Alice securely transfers coins using blind signatures from her online balance to her offline wallet. The process commences with the cryptographic hashing of the coin's public key, followed by the server's blind signing this hash. Subsequently, Alice unblinds this signature to derive the authentic signature of the coin, which she then archives as a novel electronic currency unit.

The procedure adopted for blind signatures is the one developed by David Chaum in [11] and presented in Fig. 2: The blind signature procedure, involves multiple steps to guarantee secure transactions. To commence, a client verifies himself with his commercial bank, which grants access to their device. This device acquires a public denomination key from the central bank for a specific coin value. Then, the device generates a coin's key pair alongside a blinding factor for enhanced security. After hashing and blinding the coin's public key, the device transmits this, along with an authorization for the coin withdrawal, to the commercial bank via a secure channel. The commercial bank deducts the amount from the client's account and authorizes the transaction with its own signature. This data is transmitted to the central bank, which applies a blind signature using its private key and sends it back. The client's device then unblind the signature using the blinding factor and securely stores the completed digital coin for future use.

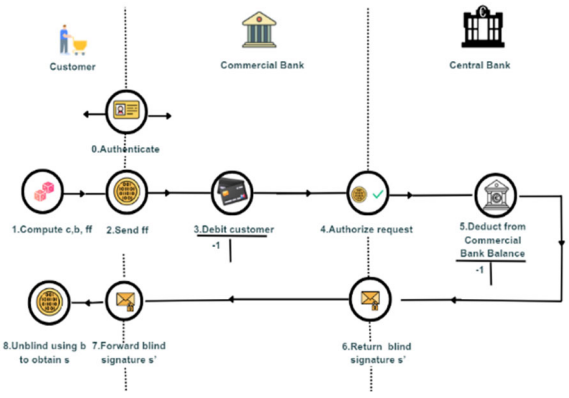

Fig. 2. CBDC withdrawal.

4.4. Offline Sealed Transaction Protocol

4.4.1. Contract Negotiation

Algorithm 2. Offline Contract Payment Transaction

Purpose: Alice wants to transfer x amount to Bob offline by negotiating a contract payment which contains a payments instruction.
Private inputs:
$\vec{w}, \vec{x}, Jk, Bob\_publickey, Contract, nonce, PK$.
Public inputs: $\boldsymbol{VkB.cert}$
Output: Encrypted transaction data
1. Bob selects $\boldsymbol{receiver \leftarrow Vk_B.cert}$
2. Send payment request to Alice.
3. Alice retrieves an electronic coin corresponding to the requested amount.
4. Alice signs the contract using the electronic coin with $\boldsymbol{SignContractUsingEcoin(s)}$.
5. Generate $\boldsymbol{\pi \leftarrow GenerateProof(\vec{w}, \vec{x}, Jk)}$.
6. Encrypt the transaction data with $\boldsymbol{Encrypt\_Transaction(\pi, Bob_{publickey}, Contract, nonce, PK)}$.
7. Send the encrypted transaction Data to Bob

Through Algorithm 2, Alice negotiates a payment contract with Bob. Bob sends her a payment request detailing the amount and including a certificate. Alice uses a digital coin to sign the payment contract. Signing a contract with a valid coin is Alice's directive to pay Bob, identified by his certified public key in the contract. If one coin isn't enough for the total amount, clients can sign a contract using multiple coins. Alice then generates a payment proof using the 'proof' function. This function produces a proof string π (comprising eight elliptic curve points) using a public input ($\vec{x}$,), a private witness $\vec{w}$, and the proving key PK. In this case, the witness is Alice's private key. Alice then sends her public key (VkA), her certified public key ($VkA.cert$), the signature (s), the zk-SNARK proof, Bob's public key ($VkB.cert$), and the nonce. The transaction is encrypted using the private key (which Alice can't access as it's in the TEE) through the hashgraph method and sent via NFC.

4.4.2. Contract Checking

Algorithm 3. Transaction Verification by Bob.

Purpose: Bob checks that the transaction is valid by verifying Alice's legitimacy as the sender of the transaction, by verifying the authenticity of the coin and by verifying the zk-SNARK proof.
Private inputs: $cert.Vk_A$, $Bob\_publickey$, Pk, $Contract, s, f, Vk, \vec{x}, \pi, Zk$.
Public inputs: None
Output: Decision bit db
1. Verify Alice's public key:
- If $\boldsymbol{CertVerify(cert.Vk_A) = False}$, then transaction fails and exit.
2. Verify coin's authenticity:
- If Validate_CoinSignature $\boldsymbol{(Bob\_publickey, Pk, Contract, s, f, Vk_A, \pi) = False}$, then transaction fails and exit.
3. Check that Bob is the receiver of the transaction:
- If $\boldsymbol{RecVerify(Vk_B) = False}$, then transaction fails and exit.
4. Verify the zk-SNARK proof:
- $\boldsymbol{db \leftarrow verify\_proof(\pi, Zk, \vec{x},)}$
- If $\boldsymbol{db = 0}$, then transaction fails.
5. If all checks pass, the transaction is valid and associated with $\boldsymbol{Bob\_publickey}$.

In Algorithm 3, Bob performs a series of checks on the validity of the transaction and the legitimacy of the sender and receiver. Bob's system records proofs and nonces to validate transactions. It rejects transactions if the proof is not associated with Bob or if duplicate proofs are received, even with different nonces, as this could indicate fraudulent attempts. If the transaction passes these checks, Bob acknowledges receipt (to Alice) and stores the coin. Then Alice confirms to Bob that she has received the acknowledgment. Fig. 3 illustrates steps 6 and 7 of the sealed transaction between Alice and Bob.

4.4.3. Coin's Destruction

Algorithm 4. Coin Destruction in Alice's Device.

Private inputs: $s, Hashed\_Coin$
Public inputs: None
Output: Acknowledgment status (Ack)
1. Upon receiving acknowledgment:
- Hash the coin: $\boldsymbol{Hashed\_Coin \leftarrow HashCoin(s)}$
2. Destroy the hash:
- $\boldsymbol{clearKey(Hashed\_Coin)}$
3. If Hashed_Coin is successfully cleared:
- Ack = "Acknowledgment Successful and Coin Destroyed"
- Else, Ack = "Acknowledgment Failed or Coin Not Destroyed"

Concerning Algorithm 4, once the actual payment has been acknowledged, the original coin is hashed and then destroyed. At the end of the procedure Bob will have the original coin's version. It's worth highlighting that in situations where Alice's coin destruction process isn't entirely thorough, the coin becomes unusable once Bob receives the confirmation of the transaction's completion. This aspect resolves the double-spending problem. This is exactly what happens with cash, because when the issuer spends his money, he loses it to the recipient.

4.4.4. Coin Deposit and Exchange

Algorithm 5. Bank Coin Verification and Exchange.

Private inputs: Client, bank, s, f, proof, C, coins(e,n)
Public inputs: None
Output: Verification status (Vstatus)
1. Monitor the bank's database for coin defunding:
- If $DepositCoin(Client, bank, s, f, proof, C) = False$:
- Vstatus = "Coin Already Defunded or Verification Failed"
- Exit.
2. For clients wishing to exchange old coins:
-$NewCoins \leftarrow ExchangeOldCoinsForNew(coins(e,n))$
- If NewCoins are successfully generated:
- $Vstatus$ = "*Coins Exchanged Successfully*"
- Else:
- Vstatus = "$Coin\ Exchange\ Failed$"

Algorithm 5 describes the process the bank use to ensure through monitoring of its database that the coins are not already defunded by performing the same verification as Bob when he receives a contract. Moreover, clients can exchange their old coins for new ones after verification. Fig. 4 illustrates the detailed transaction between the sender and the receiver using our secure offline payment protocol.

5. Discussion

5.1. Addressing User Concerns and Profiling Risks

Our solution is centered around incorporating privacy principles into CBDC infrastructures. Through the utilization of privacy-enhancing technologies (PET) that merge blind signatures with zk-SNARK, our approach safeguards client identities, transaction amounts, and detailed activities from central banks. This prevents the tracing of a coin's original user based on payment history. The recipient only possesses payment proof without sender identity or transaction metadata. However, commercial banks manage Know Your Customer (KYC) and monitor fund movements to prevent financial instability due to bank disintermediation. Users express concerns about banks knowing their expenses in detail, not their identification. Profiling risks pertain to understanding consumer spending habits, not their monetary identity.

We developed a privacy-focused system emphasizing recipient client privacy for Anti-Money Laundering (AML) and Combating the financing of terrorism (CFT). Client revenue transactions are managed by associated contracts, facilitating inspection by commercial banks and regulatory authorities for revenue anomalies. Offline transaction amounts might be capped, and withdrawn amounts shouldn't surpass spent amounts without synchronization with the online commercial bank.

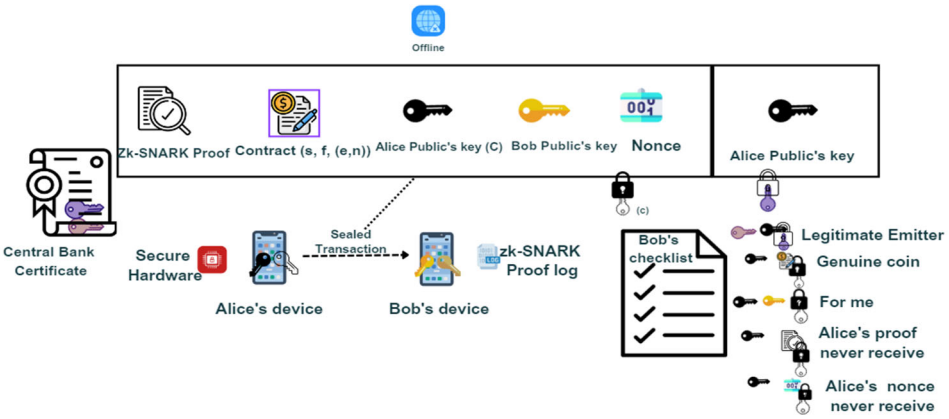

Fig. 3. Offline CBDC Transaction.

5.2. Innovative CBDC Payment Approach

Our system introduces an innovative CBDC payment approach, different from the typical bank deposit-based models. It replicates the security of banknotes in digital form, enhancing security against counterfeiting. Coins signed via signature keys can be exchanged for new ones before key expiration, mitigating security risks even in case of key compromise. if value signature keys are compromised, banks can revoke them via the app. Our open-source, software-based solution doesn't necessitate new infrastructure, prevents fraud and provides quantum-resistant privacy due to the blinding factor.

5.3. Addressing Protocol Limitations and zk-STARK Advantages

Albeit our protocol offers a proven guarantee of security and privacy protection, it can be improved by replacing zk-SNARK with zk-STARK [15], whose theoretical and experimental potential makes it an alternative and enhanced solution. Unlike

zk-SNARKs, zk-STARKs do not require an initial setup phase. This characteristic eliminates the risks associated with generating and securely storing the initial parameters, which, if compromised, could compromise the security of the entire system. The zk-STARKs are resistant to quantum attacks and provide future protection against threats from quantum computers. Moreover, zk-STARKs proofs offer faster proof generation, which is crucial in high-volume systems processing thousands of transactions per second. Nevertheless, zk-STARKs being a relatively recent technology, it is important to seriously evaluate their reliability and test them in plethora of use cases. Additionally, they are more resource-intensive considering the size of the zk-STARK proof.

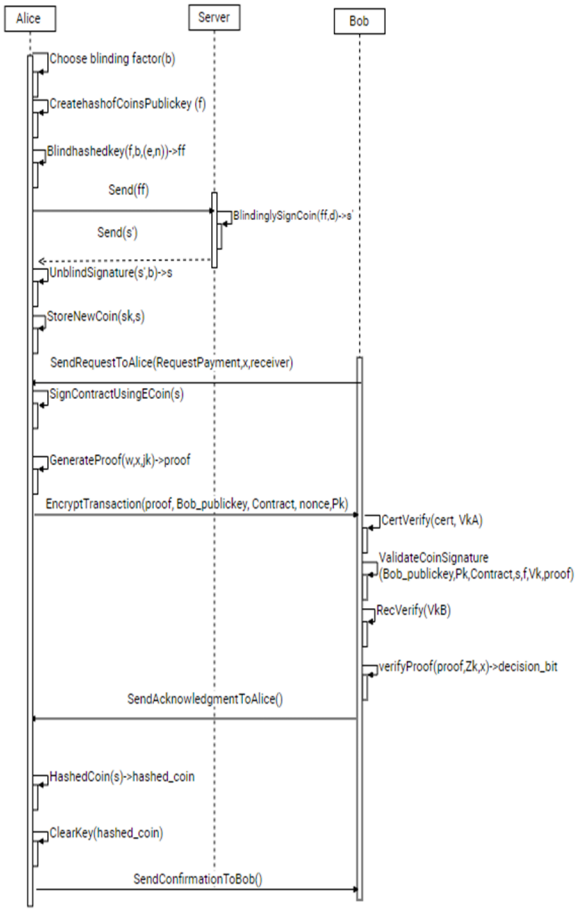

Fig. 4. A sequence diagram of the detailed transaction between the sender and the receiver using our secure offline payment protocol.

5.4. Future Integration and Expansion

Our future work involves integrating our solution into a blockchain environment, incorporating zk-SNARK payment proofs as a challenge. Then, we can also extend the use of this protocol to a smart card and consider the case of payment from an offline holder to a user who does not have one but can connect to the blockchain [16, 17] via their identifier and carry out defunding.

6. Conclusions

All in all, the proposed protocol ensures end-to-end data privacy in CBDC offline payment, even in the event of a quantum attack, without compromising security requirements. With the expenditure of signed coins, our solution provides the same experience as cash and does not require a new infrastructure to deploy. Indeed, it could also be blockchain compatible. In terms of perspectives, the zk-SNARK can be replaced by the Zero-knowledge Scalable Transparent Argument of Knowledge (zk-STARK) which has the advantage of offering faster calculations and protecting against post-quantum attacks. Indeed, in order to improve our solution compared to cash, we are considering offline programmability as a first step towards smart contracts.

Acknowledgements

The authors express their gratitude for the valuable input and constructive feedback given by the following organizations: Fime and Greyc Laboratory.

References

[1]. European Central Bank, Eurosystem, https://www.ecb.europa.eu/pub/pdf/other/Eurosystem_report_on_the_public_consultation_on_a_digital_euro~539fa8cd8d.en.pdf

[2]. R. Auer, R. Boehme, The technology of retail central bank digital currency, *BIS Quartely Review*, 2020.

[3]. M. Christodorescu, et al, Towards a Two-Tier Hierarchical Infrastructure: An Offline Payment System for Central Bank Digital Currencies, *arXiv Preprint*, 2012, arXiv:2012.08003.

[4]. A. Dogan, et al, Smart Card Based Offline Payment System for Central Bank Digital Currencies, in *Proceedings of the Conference on Blockchain and Cryptocurrency (B2C'22)*, Barcelona, Spain, 9-11 November 2022, pp. 114-127.

[5]. L. Mainetti, et al, A Sustainable Approach to Delivering Programmable Peer-to-Peer Offline Payments, *Sensors,* Vol. 23, Issue 2, 2023, 1336.

[6]. Offline CBDC payments, *IDEMIA*, 2023.

[7]. G. Samid, A LeVeL Paying Field: Cryptographic Solutions towards Social Accountability and Financial Inclusion, *Cryptology ePrint Archive*, 2022, 2022/130.

[8]. M. Adams, et al, An integrated approach for electronic identification and central bank digital currencies, *Journal of Payments Strategy & Systems*, Vol. 15, Issue 3, 2021, pp.287-305.

[9]. GlobalPlatform, https://globalplatform.org/wp-content/uploads/2018/04/GlobalPlatform_TEE_White paper_2015.pdf

[10]. D. Chaum, et al., How to issue Central Bank Digital Currency, *SNB Working Paper*, 2021.

[11]. D. Chaum, T. Moser, eCash 2.0, *SNB Working Paper*, 2022.

[12]. N. Bitansky, et al., From Extractable Collision Resistance to Succinct Non-Interactive Arguments of Knowledge, and Back Again, in *Proceedings of the 3rd*

Conference on Innovations in Theoretical Computer Science (ITCS'12), New York, United States, 08 January 2012, pp. 326-349.

[13]. M. Bertaccini, A Guide to Algorithms in Blockchain, Quantum Cryptography, Zero-Knowledge Protocols, and Homorphic Encryption, *Packt Publishing*, 2022.

[14]. Y. Chu, et al., Review of Offline Payment Function of CBDC Considering Security Requirements, *Applied Sciences*, Vol. 12, 2022, 4488.

[15]. T. Ashur, S. Dhooghe, MARVELlous: a STARK-Friendly family of cryptographic primitives, *Cryptology ePrint Archive*, 2018, 2018/046.

[16]. H. Moudoud, S. Cherkaoui, L. Khoukhi, Towards a Scalable and Trustworthy Blockchain: IoT Use Case, in *Proceedings of the conference on IEEE International Conference on Communications (ICC'20)*, Montreal, Canada, 14-23 June 2021, pp. 1-6.

[17]. Z. A. E. Houda, A. Hafid, L. Khoukhi, BrainChain – A Machine learning Approach for protecting Blockchain applications using SDN, in *Proceedings of the conference on IEEE International Conference on Communications (ICC'20)*, Dublin, Ireland, 7-11 June 2020, pp. 1-6.