Information Sciences 625 (2023) 157–174

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Traceable one-time address solution to the interactive blockchain for digital museum assets

Liutao Zhao [a,c], Lin Zhong [c], Jiawan Zhang [a,b,*]

[a] College of Intelligence and Computing, Tianjin University, Tianjin 300350, China
[b] Tianjin Cultural Heritage Conservation and Inheritance Engineering Technology Center and Key Research Center for Surface Monitoring and Analysis of Relics, State Administration of Cultural Heritage, Tianjin 300350, China
[c] Beijing Computing Center, Beijing 100094, China

ARTICLE INFO

Article history:
Received 23 July 2022
Received in revised form 30 October 2022
Accepted 29 December 2022
Available online 5 January 2023

Keywords:
Blockchain
Supervisable
One-time address
Anonymity
Traceability

ABSTRACT

Blockchain systems often use public keys or addresses as pseudonym accounts to protect the identity of users. However, as the blockchain system is transparent, an adversary can analyze all the public keys or addresses and obtain some real information about users, making the pseudonym mechanism ineffective. CryptoNote V2.0 is the first blockchain system to introduce a one-time address technology to enable the true anonymity of a user. But it does not have rigorous security proof. Moreover, due to application requirements, for one thing, users need anonymity. For another, administrators need to trace the identity of anonymous users. Therefore, a traceable one-time address scheme for the interactive system of digital museum items is proposed to fulfill these concerns. It allows the recipient to receive transactions anonymously, and a supervisor can trace the identity of a user efficiently. Two security models are defined and two corresponding instances are constructed. We prove that our instances are secure in IND-PK-CPA/IND-PK-CCA mode, respectively. Analysis and experiments show that the proposed schemes are effective in tracing the long-term identity of a recipient. Finally, the challenges of user privacy protection are enhanced with the advancement of the integration of digital museum asset interactions with blockchain. We build a blockchain system for Digital Museum assets and implement our two scheme instances in it. The system can integrate museum heritage resources around the world, share data based on blockchain, provide a large number of interactions and collections, and achieve large-scale exhibitions.

1. Introduction

In 2010, a museum data interaction project, which includes nine museums in the United States as well as OCLC (Online Computer Library Center) research, was established with funding from the Andrew W. Mellon Foundation. This project uses the CDWA Lite data-sharing technique and models the data exchange process by creating research aggregations. CDWA Lite is an XML format based on CDWA and CCO for describing core records for works of art and material culture. CDWA Lite records were designed to contribute to union catalogs and other repositories using the Open Archives Initiative harvesting protocol. The OCLC is a nonprofit membership organization that encourages libraries worldwide to work together. OCLC ser-

* Corresponding author at: College of Intelligence and Computing, Tianjin University, Tianjin 300350, China.
E-mail addresses: zhaolt@tju.edu.cn (L. Zhao), zhonglin@bcc.ac.cn (L. Zhong), jwzhang@tju.edu.cn (J. Zhang).

https://doi.org/10.1016/j.ins.2022.12.084
0020-0255/

vices are used by over 54,000 libraries in 109 countries to search, purchase, categorize, lend, and preserve print and electronic library holdings. Anyone can access the bibliographic, abstract, and full-text databases of OCLC. The interaction of digital museum assets could change cultural heritage management and even the distribution of cultural resources worldwide [1]. Thus, museums should strengthen their cooperation for digital collection resources complementation and sharing and then select and display collections in different themes. Designing exhibitions on an unprecedented scale through big data sharing allows for the integration of heritage resources worldwide and enables a variety of interpretations and presentations of collections [2].

Continuous research into the integration of digital museum asset interaction with blockchain still revealed shortcomings in the widespread application of blockchain, especially the challenges for user identity and privacy protection. Blockchain provides privacy protection for the identities of traders by using public keys or addresses as pseudonyms and account numbers of users. However, the pseudonym mechanism offers only weak anonymity [3]; together with the transparency of blockchain transactions, anyone can draw information regarding the transactions of a user by analysis and observation [4].

A pseudonym is an anonymous certificate system that does not provide any information regarding the true identity of a user. On-demand pseudonyms can be produced or pre-generated. Numerous studies on inferring private information regarding transaction users based on blockchain transaction records have emerged. For instance, Androulaki et al. [5] designed a simulation experiment to match blockchain addresses to the physical identities of users, which essentially achieved an accuracy rate of 42% through clustering techniques of transactional user behavior.

Monaco [6] developed an analytical model based on 12 parameters, such as transaction interval and amount flow, to analyze the transaction pattern of users. This model successfully recognized the real identity of users with up to 62% accuracy. Meanwhile, decentralization of the blockchain bypasses the regulation of existing organizations or institutions. For regulatory authorities, identifying the owner of each transaction account on blockchain systems and all its relevant accounts is necessary, whereas available anonymity mechanisms complicate the establishment of such an association. The Bitcoin blockchain is a public ledger, as suggested by its name.

Marian [7] encouraged users to mark their identification numbers to increase the probability of successfully detecting and sanctioning suspicious users of illegal transactions. A suspicious transaction increases the apprehension or suspicion of a reporting entity regarding the transaction due to its unusual character or circumstances or the person or group of people participating in the transaction. Anonymity and traceability are the most essential and central features of the blockchain mechanism; thus, regulation at the legal system level cannot fundamentally prevent the risk of blockchain systems. The private blockchain is an example of a centralized blockchain system because it is controlled by a single group or organization. By contrast, a public blockchain is decentralized. The key to developing and promoting blockchain applications for digital museum collections is technically enabling user identity tracking for anonymous blockchain addresses and allowing regulators to supervise blockchain transactions effectively [8]. Anonymous identity ensures the unlinkability and anonymity of the transactions of genuine users while also making those considered malicious or doubtful accordingly.

CryptoNote V2.0 [9] is the first one-time address technology. This technology generally enables a sender to initiate a transaction to a one-time address. The scheme includes the following four steps. First, the sender uses the long-time public key of the receiver to calculate a one-time beneficiary address and pays some amount to the one-time address. Second, the sender sends the one-time address to the receiver. Third, the receiver calculates a correct one-time private key corresponding to the one-time beneficiary address. Fourth, the receiver can take a ring signature based on the one-time private key to obtain payment in this transaction system. Therefore, Bob can receive a payment associated with a one-time address that is unlinkable to any adversary or supervisor.

However, in supervisable blockchain systems [10,11], protecting not only the identity privacy of receivers but also tracing their identity is necessary [12]. Therefore, a traceable scheme is proposed for a one-time address generation of blockchain systems. The proposed scheme inherits the advantage of the one-time address technology of CryptoNote V2.0, that is, both parties can independently generate the one-time address to realize the anonymous receiver. The property of identity tracing for the receiver, which is essential in supervisable blockchain systems, is also realized.

Our contributions. A traceable one-time address scheme is proposed in this paper to provide anonymity for receivers and traceability for the system supervisor in blockchain systems. In the proposed scheme, the anonymity of receivers is equal to CryptoNote V2.0 because each one-time address of a receiver appeared only once. Moreover, a supervisor who can trace the identity of each receiver exists in blockchain systems, which is different from CryptoNote V2.0. CryptoNote is a cryptocurrency application layer technology that attempts to tackle certain Bitcoin difficulties. A security model is defined for the traceable one-time address scheme, and its security in the security model is strictly proven. The proposed scheme is implemented in C programming language; its speed is also tested and compared with CryptoNote V2.0.

Specifically, the following contributions are presented.

- A traceable one-time address scheme is introduced. The core principle of the proposed scheme is that the sender embeds long-time public keys of the receiver and the supervisor when generating a one-time address for a receiver. Therefore, the receiver can accurately compute the corresponding one-time private key by using the one-time public key and long-time private key. The system supervisor can also trace the identity of the receiver.
- The proposed one-time address scheme is similar to public encryption schemes, such as ElGamal encryption; thus, we define a security model in a similar way. The proposed two scheme instances are proven to be secure in the defined security models.

- The traceable one-time address scheme is implemented and its performance is evaluated and compared with CryptoNote V2.0. Besides, analyses and experimental test results show that the proposed scheme is efficient in tracing the identity of an anonymous user. Finally, We build a blockchain system for Digital Museum assets and implement our two scheme instances in it.

Paper organization. The remainder of this paper is organized as follows. Section 2 reviews some related works. Section 3 represents an alliance chain architecture for digital museum asset trading. Section 4 provides some preliminaries. Section 5 presents two concrete instances for the traceable one-time address scheme. Section 6 formally defines a security model as well as strict security proof. Section 7 compares the proposed scheme with the one-time address of CryptoNote V2.0 in function and computational complexity. Section 8 gives some experimental test results of the proposed scheme and the one-time address scheme of CryptoNote V2.0. Section 9 concludes the paper.

2. Related works

In blockchain systems, there are 4 typical technologies in identity privacy protection, including Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARKs), mixed coin protocols, ring signatures, and one-time address technology.

zk-SNARKs [13–18] and some schemes [19–24] were proposed to achieve anonymous payments. In Zerocoin, validators do not have to check the validity of a digital signature in each transaction but must validate whether a payment belongs to a list of valid payments. The origin of the payment cannot be unlinked from previous transactions, thus preventing transaction graph analyses. However, the destination and the amounts of payments can still be revealed. Zerocash was then proposed to address the destination and leakage problems of amounts. The cryptographic components of Zerocash must be properly implemented to verify that the zk-SNARKs, which is the "heaviest" component, are sufficiently efficient in reality. The Zerocash uses the zk-SNARKs to hide transaction amounts and the destination. However, zk-SNARKs has relatively high computational complexity and long proof compared with the one-time address. Traceability is also not implemented in zk-SNARKs.

Mixed coin protocols [25] provide privacy protection by transferring payments from an input set of transaction addresses to an output set of transaction addresses such that linking the input with the output address is difficult. Mixcoin [26] provides mixing services by adding a trusted third party. However, the trusted third party may be malicious because one can steal coins of users and destroy their privacy. The theft behavior can be detected by users but cannot be prevented. Blindcoin [27] improves on Mixcoin by preserving the privacy of users against the mixing service. However, similar to Mixcoin, theft cannot still be prevented. If two-thirds of the honest third parties are really honest, then CoinParty [28] is secure. CoinParty is secure as long as two-thirds of the mixing parties are honest, but the theft problems remain unsolved. Finally, the problem of coin theft was solved by CoinJoin [29] and CoinShuffle [30].

CoinShuffle is a decentralized Bitcoin mixing technology that enables users to use Bitcoin anonymously. CoinJoin is a method for facilitating anonymous Bitcoin transactions over the internet. CoinJoin is a multiparty Bitcoin transaction, in which all participants put in and get the same amount of Bitcoin. However, the addresses are disorganized in the transaction, complicating the tracing of the origin of Bitcoins. Similar to CoinShuffle, Meiklejohn et al. [31] provided rigorous proof of anonymity for their scheme. Overall, the assumption of a trusted third party is too strong, which is unfit in blockchain systems. Moreover, the bottlenecks of system performance include computing power and scheduling capability of the trusted third party. Furthermore, anonymity in mixing technologies is relatively low because an adversary can conduct data analysis to brute force cracking. However, a polynomial-time adversary cannot break the proposed scheme.

Monero and many other ring signature schemes [32–38] are proposed to achieve signer anonymity and signature linkability. In these ring signature schemes, validators must use multiple public keys to validate a signature without knowing which public key belongs to the real signer. However, if a signer generates two signatures using the same transaction, then the two signatures will be linked. By contrast, Saberhagen Van Saberhagen [9] proposed a one-time address technology to hide transaction receivers which has been elaborated on in the Introduction. However, their scheme lacks a security model and formal security proof compared with the proposed scheme.

In blockchain systems, linkable ring signatures are insufficient because a supervisor needs to trace each sender. Therefore, the traceable ring signature scheme [39] was proposed. A traceable ring method is similar to a ring signature, with the exception that it can limit "extreme" anonymity. The traceable ring signature features a tag that includes a list of ring members as well as issues, such as a social gathering or an election. Similarly, the supervisor cannot trace each receiver in the one-time address technology. Therefore, the traceable one-time address scheme is proposed to achieve identity privacy protection and supervision of the receiver.

3. System architecture

Alliance chain architecture for digital museum asset trading is represented in Fig. 1, in which two museums, such as A and B, are considered. The data exchange between museums A and B takes place on the museum data exchange platform. First, the content is created from museum A. The write-in new block is then transferred to the data exchange platform. Furthermore, the distributed storage platform transfers the digital asset filing and returns rights filing from the regulator of digital

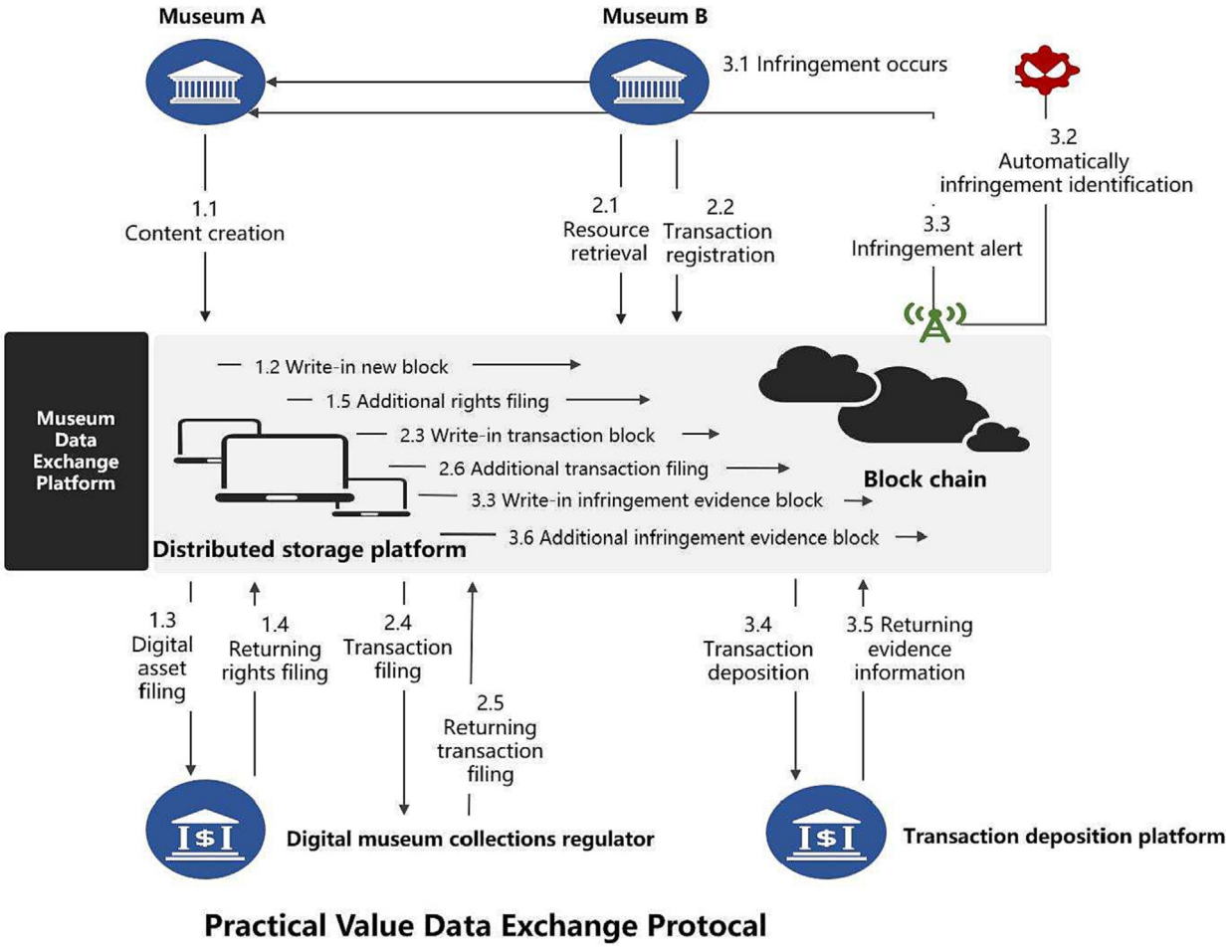

Fig. 1. Alliance chain architecture for digital museum assets trading.

museum collections. Resource retrieval and transaction regulation are also transferred from museum B to the data exchange platform. Distributed storage offers a unique disaster recovery policy for applications of users. The adoption of a fully hybrid cloud is easy with a highly available storage cluster that spans different data centers or clouds. The alliance chain architecture of the digital museum is defined by considering two museums A and B, which will share data via a data exchange platform. As used by the museum, a distributed storage platform is an architecture that can spread data over numerous physical servers and typically across different data centers. A digital transaction is a method of conducting business without the use of cash. A digital transaction entails the cooperation of numerous stakeholders, including significant financial institutions and a variety of economic sectors. The digital museum collection regulator is used for the transaction process with the intermediate of the distributed storage system.

1.1–1.5: Process of ownership confirming for digital museum collections
2.1–2.6: A trading process of access rights for digital museum collections
3.1–3.6: Infringement deposition process for digital museum collections

- The roles of users include museums, token institutions, and regulators.
- Regulator: assessing and enforcing access and regulating data and transactions. The platform builder selects a body with the appropriate credibility/ qualifications as a regulator based on the law.
- Museum: uploading data, saving tokens and executing transactions, and exchanging.
- Token institution: issuing trading tokens.

4. Preliminaries

Before presenting the traceable one-time address scheme, a few concepts and a scheme related to the proposed one are first reviewed.

- **Discrete Logarithm Problem.** Given any two group elements of a group $G, G\prime \in \mathbb{G}$, finding an integer $n \in \mathbb{Z}^*$ is intractable such that the following equation holds:

  $$G\prime = n \cdot G$$

- **Computational Diffie-Hellman Problem.** Given three group elements of a group $G, a \cdot G, b \cdot G \in \mathbb{G}, a, b \in \mathbb{Z}$, finding an element $G\prime \in \mathbb{G}$ is intractable for any polynomial-time algorithm such that the following equation holds:

  $$G\prime = ab \cdot G$$

- **Decision Diffie-Hellman Problem.** Given four group elements $G, a \cdot G, b \cdot G, c \cdot G \in \mathbb{G}$, outputting a correct judgment is intractable for any polynomial-time algorithm: if $c = ab$, then True else False is outputted. If True is outputted, then the quadruple $(G, a \cdot G, b \cdot G, c \cdot G)$ are DDH tuples.

Fig. 2 shows the process of the one-time address generation. Alice and Bob can compute the same Diffie-Hellman session key by using their private message, respectively. Alice computes the one-time address with the shared secret key and a random number r. She sends the address $P\prime$ and a random elliptic curve point R to Bob.

1. A sender (Alice), wants to take a transaction to a receiver, called Bob. Then, Bob sends his long-time public key $\mathrm{PK} = (A, B)$ to Alice.
2. Alice selects a random number $r \in [1, l-1]$ and computes a one-time public key as follows

   $$P := \mathrm{hash}(r \cdot A) \cdot G + B$$

3. Alice uses P as the one-time public key for the output and an auxiliary value $R = r \cdot G$. The one-time address $Addr^{\mathrm{OneTime}}$ of Bob comprises a one-time public key P and an auxiliary value R

   $$\mathrm{Addr}_{Bob}^{\mathrm{OneTime}} = (P, R)$$

   Notably, each receiver has different long-time public keys $(A_i, B_i), i = 1, \ldots n$. The sender can even use the same random number r to create many different one-time addresses for different receivers

   $$P_i := \mathrm{hash}(r \cdot A_i) \cdot G + B_i, i = 1, \ldots n$$

   However, considering multiple transactions to the same receiver, the sender must use a different random number $r\prime$.

   $$P\prime := \mathrm{hash}(r\prime \cdot A) \cdot G + B$$

   Otherwise, the generated one-time address will be repeated $P\prime = P$.
4. Alice broadcasts the transaction to the blockchain system, and sends the auxiliary value R to Bob.
5. Fig. 3 shows that Bob can compute the one-time private key $\mathrm{sk}^{\mathrm{OneTime}}$ by using his first half of the long-time private key a

   $$P' := \mathrm{hash}(a \cdot R) \cdot G + B$$

   As the following equation holds

   $$a \cdot R = ar \cdot G = r \cdot A$$

   Bob can compute the corresponding one-time public key $P\prime = P$ correctly.
6. Bob takes the corresponding one-time private key as input and the second part of his long-time private key and computes as follows:

   $$\mathrm{sk}^{\mathrm{OneTime}} := \mathrm{hash}(a \cdot R) + b$$

   Thus, he can check the following equation

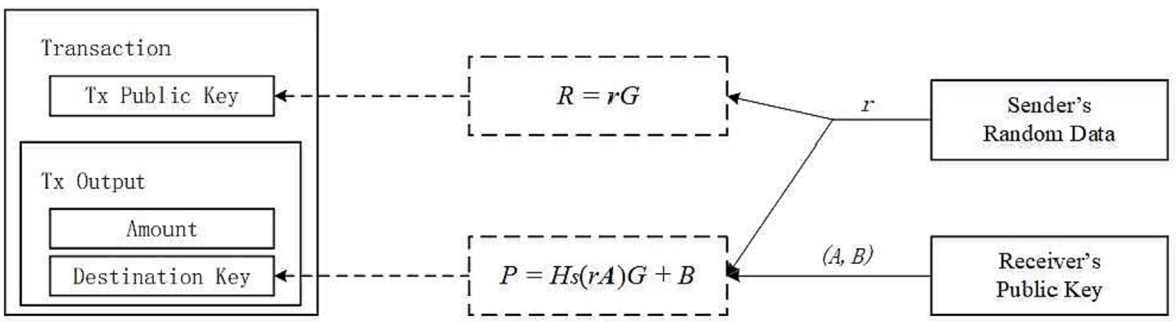

Fig. 2. CryptoNote V2.0 one-time address generation model.

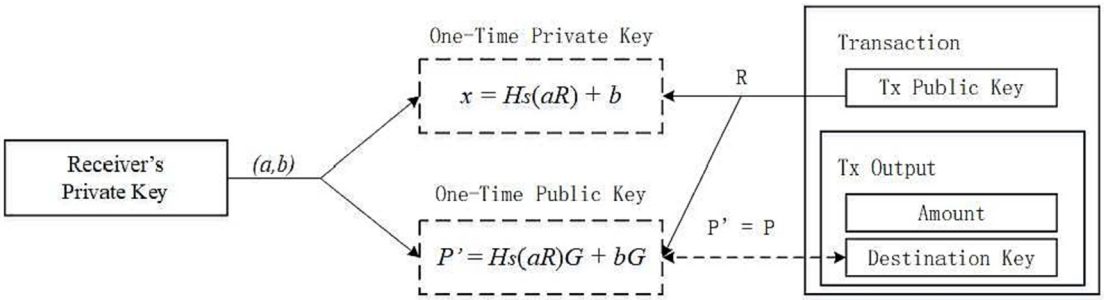

Fig. 3. CryptoNote V2.0 one-time private key generation model.

$P = \mathrm{PK}^{\mathrm{OneTime}} = \mathrm{sk}^{\mathrm{OneTime}} \cdot G$.

Therefore, he can take payment at any time by signing a transaction using this one-time private key $\mathrm{sk}^{\mathrm{OneTime}}$.

Algorithm 1: one-time address generation process

Input: sender = Alice, receiver = Bob,
  one-time address $\mathrm{Addr}^{\mathrm{OneTime}}$,
  long-time public key $\mathrm{PK} = (A, B)$.
select randomly $r \in [1, l-1]$
generate a one-time public key $P := \mathrm{hash}(r \cdot A) \cdot G + B$
Compute auxiliary value $R := r \cdot G$
For each $(A_i, B_i), i = 1, \ldots n$
  Evaluate the one-time address of the receiver
  $\mathrm{Addr}_{Bob}^{\mathrm{OneTime}} = (P, R)$
  create a one-time address for different receivers
  $P_i := \mathrm{hash}(r \cdot A_i) \cdot G + B_i, i = 1, \ldots n$
  If(multiple transactions → same receiver)
    $P\prime := \mathrm{hash}(r\prime \cdot A) \cdot G + B$
    Else
    generate one-time address $P\prime = P$
  End if
End for
For each receiver
  If(private input = one-time private key)
  Compute $\mathrm{sk}^{\mathrm{OneTime}} := \mathrm{hash}(a \cdot R) + b$
  Check $P == \mathrm{PK}^{\mathrm{OneTime}} == \mathrm{sk}^{\mathrm{OneTime}} \cdot G$
  Transaction using the one-time private key $\mathrm{sk}^{\mathrm{OneTime}}$
  end if
end for

5. Proposed schemes

Two instances are presented for the traceable one-time address scheme.

5.1. First instance

The concrete construction of the first traceable one-time address scheme, namely **Setup**, **KeyGen**, **OnetimeAddrGen**, **OnetimeSKGen**, **IdentityTracing**, **SenderStatistics**, and **ReceiverStatistics**, is as follows.

- **Setup.** The elliptic curve equation is $y^2 = x^3 + ax + b$, where O is the identity element of the curve, G is a base point of prime order on the elliptic curve, and n is the multiplicative order of the point G. The order n of the base point G must be prime. Let $\mathbb{G}$ be the elliptic curve group. Assuming (that) any element of the ring Z/nZ is invertible is reasonable; therefore, Z/nZ must be a field. Hash function $\text{hash}_1 : \{0,1\}^* \to Z_n, \text{hash}_2 : \mathbb{G} \to Z_n$. The system parameters are

$$\text{SP} = (a, b, \mathbb{G}, G, n, \text{hash}_1, \text{hash}_2)$$

- **KeyGen.** User 1, a sender of the system, randomly chooses $a_1, b_1 \in Z_n^*$, and computes as follows

$$A_1 := a_1 \cdot G,$$
$$B_1 := b_1 \cdot G$$

The long-time private key of the sender is $\text{sk}_1 = (a_1, b_1)$, and his long-time public key is $\text{PK}_1 = (A_1, B_1)$.
Similarly, user 2 is a receiver of the system, in which the long-time private key is $\text{sk}_2 = (a_2, b_2)$, and his long-time public key is $\text{PK}_2 = (A_2, B_2)$. The following is required:

$$A_2 = a_2 \cdot G,$$
$$B_2 = b_2 \cdot G$$

User 3 is a supervisor of the system, in which the long-time private key is $sk_3 = (a_3, b_3)$, and his long-time public key is $\text{PK}_3 = (A_3, B_3)$. The following is required:

$$A_3 = a_3 \cdot G,$$
$$B_3 = b_3 \cdot G$$

All these long-time public keys can be linked with their identities.

- **OnetimeAddrGen.** The sender takes as input a random number $r_1 \in Z_n^*$, the first part of his long-time private key a_1, the long-time public key of the receiver $\text{PK}_2 = (A_2, B_2)$, the first part of the long-time public key A_3 of the supervisor, and computes as follows

$$r_2 := \text{hash}_1(r_1, a_1 \cdot A_2),$$
$$r_3 := \text{hash}_2(r_2 \cdot A_3),$$
$$R := r_2 \cdot G,$$
$$\text{PK}_2^{\text{OneTime}} := r_3 \cdot G + R + B_2$$

Let $\text{Addr} = (\text{PK}_2^{\text{OneTime}}, R, r_1)$ be the one-time address of the receiver, where (R, r_1) are auxiliary information, and $\text{PK}_2^{\text{OneTime}}$ is the one-time public key of the receiver. The sender can pay to this one-time address of the receiver, and sends the first part of his long-time public key A_1 to the receiver privately. In each transaction, if the receiver uses a new one-time address Addr, then an adversary cannot trace his long-time public key in polynomial time. A probabilistic polynomial-time algorithm can only perform a polynomial number of operations, including at most a polynomial number of coin flips. The above calculation is dual. The receiver can also generate his one-time address on his own. Specifically, the receiver takes as input a random number $r_1\prime \in Z_n^*$, his long-time private key $sk_2 = (a_2, b_2)$, the first part of the long-time public key A_1 of the sender, the first part of the long-time public key A_3 of the supervisor, and computes as follows

$$r_2\prime := \text{hash}_1(r_1\prime, a_2 \cdot A_1),$$
$$r_3\prime := \text{hash}_2(r_2\prime \cdot A_3),$$
$$R := r_2\prime \cdot G,$$
$$\text{PK}_2^{\text{OneTime}} := r_3\prime \cdot G + R + B_2$$

Therefore, the sender and the receiver can generate the one-time address. However, only the receiver can accurately generate the corresponding one-time private key. Additional information will be provided later.

- **OnetimeSKGen.** The receiver takes as input his long-time private key $\text{sk}_2 = (a_2, b_2)$, the one-time address $\text{Addr} = (\text{PK}_2^{\text{OneTime}}, R, r_1)$, the first part of the long-time public keys of the receiver A_1 and the supervisor A_3, and computes as follows

$$r_2 := \text{hash}_1(r_1, a_2 \cdot A_1),$$
$$r_3 := \text{hash}_2(r_2 \cdot A_3),$$
$$\text{sk}_2^{\text{OneTime}} := r_3 + r_2 + b_2$$

The receiver accepts the transaction if the following two equations hold

$$R == r_2 \cdot G$$
$$\text{PK}_2^{\text{OneTime}} == \text{sk}_2^{\text{OneTime}} \cdot G$$

These two checks ensure that R and $\text{PK}_2^{\text{OneTime}}$ are correct in the one-time address Addr. Therefore, $\text{sk}_2^{\text{OneTime}}$ computed by the receiver is the correct one-time private key corresponding to the one-time public key $\text{PK}_2^{\text{OneTime}}$.

- **IdentityTracing.** The supervisor takes as input the first part of his long-time private key a_3 and the one-time address $\text{Addr} = (\text{PK}_2^{\text{OneTime}}, R, r_1)$, and computes as follows

$$r_3 := \text{hash}_2(a_3 \cdot R),$$
$$B_2 := \text{PK}_2^{\text{OneTime}} - R - r_3 \cdot G$$

  Therefore, the supervisor can trace the long-time public key B_2 of the receiver by using the one-time address Addr and his long-time private key a_3.
- **SenderStatistics.** The sender according to the trade of digital signature to find the one-time address $\text{Addr} = (\text{PK}_2^{\text{OneTime}}, R, R_1)$, takes as input his private key a_1, the long-term public key of the intended recipient, (a_2, B_2). The local long-term public key of the regulator A_3 is calculated as follows

$$r_2 := H_1(r_1, a_1 \cdot A_2),$$
$$r_3 := H_2(r_2 \cdot A_3)$$

  Check if the following equation is holds

$$R == r_2 \cdot G$$
$$B_2 == \text{PK}_2^{\text{OneTime}} - r_3 \cdot G - R$$

  If yes, accept, otherwise reject. Therefore, the sender can use his long-term private key a_1 and the one-time address Addr to verify whether the long-term public key B_2 of the target receiver is correct. Therefore, he can trace the identity of the receiver and realize the identity statistics of recipients.
- **ReceiverStatistics.** The receiver takes as input his long-time private key $a_2$and his long-time public key B_2, the long-time public key $a_1$of the sender, the long-time public key $a_3$of the supervision, and the one-time address $\text{Addr} = (\text{PK}_2^{\text{OneTime}}, R, R_1)$, computes as follows

$$r_2 := H_1(r_1, a_2 \cdot A_1),$$
$$r_3 := H_2(r_2 \cdot A_3),$$

  Check if the following equation is holds

$$R == r_2 \cdot G$$
$$B_2 == \text{PK}_2^{\text{OneTime}} - r_3 \cdot G - R$$

  If yes, then accept, otherwise reject. Therefore, the receiver can use his long-time private key a_2 and one-time address Addr to verify whether the long-time public key of the sender is A_1. Therefore he can trace the identity of the sender and realize the identity statistics of senders.

Theorem 1. *Only the corresponding receiver can compute correctly the one-time private key according to the one-time public key generated by the sender.*

Proof 1. The expression of the two random numbers computed by the sender is as follows

$$r_2 := \text{hash}_1(r_1, a_1 \cdot A_2) = \text{hash}_1(r_1, a_1 a_2 \cdot G),$$
$$r_3 := \text{hash}_2(r_2 \cdot A_3)$$

By contrast, the two random numbers computed by the receiver are as follows

$$\tilde{r}_2 := \text{hash}_1(r_1, a_2 \cdot A_1) = \text{hash}_1(r_1, a_2 a_1 \cdot G),$$
$$\tilde{r}_3 := \text{hash}_2(r_2 \cdot A_3)$$

Therefore, the following equations hold

$$r_2 == \tilde{r}_2$$
$$r_3 == \tilde{r}_3$$

Consequently, the sender and the receiver can compute the same random numbers from different angles, because r_1 is a public component in the one-time address Addr.

The one-time public key of the receiver computed by the sender is as follows

$$\text{PK}_2^{\text{OneTime}} := r_3 \cdot G + R + B_2$$

The one-time private key of the receiver computed by the receiver is as follows

$$\text{sk}_2^{\text{OneTime}} := r_3 + r_2 + b_2$$

Therefore, the following equation is presented

$$\begin{aligned} &\mathrm{sk}_2^{\mathrm{OneTime}} \cdot G = (r_3 + r_2 + b_2) \cdot G \\ &= r_3 \cdot G + R + B_2 = \mathrm{PK}_2^{\mathrm{OneTime}} \end{aligned}$$

Thus, the one-time private key generated by the receiver corresponds to the one-time public key generated by the sender.

Moreover, as the one-time private key $\mathrm{sk}_2^{\mathrm{OneTime}}$ is the sum of r_3, r_2 and b_2, although the sender and the supervisor compute r_3, they cannot compute the second part of the long-time private key of the receiver b_2. Therefore, only the receiver can compute $\mathrm{sk}_2^{\mathrm{OneTime}}$, which means the amount paid to the one-time address Addr can only be spent by the receiver. This completes the proof of Theorem 1. □

Algorithm 2: one-time private key of the receiver

```
Input: one-time private key sk_2^OneTime
Output: one-time public key generated by the sender
For each sender
  Compute random numbers r_2, r_3
  For each receiver
    Compute random number r̃_2, r̃_3
    If (r_2 == r̃_2, r_3 == r̃_3)
      one-time public key → sender
      PK_2^OneTime := r_3 · G + R + B_2
      one-time private key → receiver
      sk_2^OneTime := r_3 + r_2 + b_2
    End if
  End for
End for
```

Theorem 2. *The supervisor can trace correctly the long-time public key, that is, identification of the receiver from the one-time public key by using the first part of his long-time private key a_3.*

Proof 2. The expression of r_3 and R computed by the sender is as follows

$$r_3 = \mathrm{hash}_2(r_2 \cdot A_3) = \mathrm{hash}_2(r_2 a_3 \cdot G)$$

By contrast, the expression of r_3 can also be computed by the supervisor, because R is a public component in the one-time address Addr.

$$\tilde{r}_3 = \mathrm{hash}_2(a_3 \cdot R) = \mathrm{hash}_2(a_3 r_2 \cdot G)$$

Therefore, we have

$$\tilde{r}_3 = r_3$$

The long-time public key of the sender computed by the supervisor is

$$B_2 := \mathrm{PK}_2^{\mathrm{OneTime}} - R - r_3 \cdot G$$

This completes the proof of Theorem 2. □

Algorithm 3: traceability of the supervisor

Input: long-time private key a_3
Output: identity the receiver from the one-time public key
For each transaction
  Evaluate $r_3 := \text{hash}_2(r_2 \cdot A_3) = \text{hash}_2(r_2 a_3 \cdot G)$
  Compute one-time address Addr
  $\tilde{r}_3 := \text{hash}_2(a_3 \cdot R) = \text{hash}_2(a_3 r_2 \cdot G)$
  If$(\tilde{r}_3 == r_3)$
    The long-time public key of the sender computed by the supervisor
    $B_2 := \text{PK}_2^{\text{OneTime}} - R - r_3 \cdot G$
  End if
End for
Return: completes the proof

5.2. Second instance

The repetition with the above scheme is omitted and started directly with the one-time address generation step.

- **OneTimeAddrGen.** The sender takes as input two random numbers $r_1, \rho \in Z_n^*$, the first part of his long-time private key a_1, the long-time public key of the receiver $\text{PK}_2 = (A_2, B_2)$,and the first part of the long-time public key A_3 of the supervisor. The following computation is presented:

$$\begin{aligned} r_2 &:= \text{hash}_1(r_1, a_1 \cdot A_2), \\ r_3 &:= \text{hash}_2(r_2 \cdot A_3), \\ y_1 &:= \text{hash}_2(r_3 \cdot G) \oplus \rho, \\ R &:= r_2 \cdot G, \\ \text{PK}_2^{\text{OneTime}} &:= \rho \cdot G + R + B_2 \\ y_2 &:= \text{hash}_1(r_3, y_1, \rho, B_2, \text{PK}_2^{\text{OneTime}}) \end{aligned}$$

Let $\text{Addr} = (\text{PK}_2^{\text{OneTime}}, R, y_1, y_2, r_1)$ be the one-time address of the receiver, where (R, y_1, y_2, r_1) are auxiliary information, and $\text{PK}_2^{\text{OneTime}}$ is the one-time public key of the receiver. The sender can pay to this one-time address of the receiver, and sends the first part of his long-time public key A_1 to the receiver privately. In each transaction, if the receiver uses a new one-time address Addr, then an adversary cannot trace his long-time public key in polynomial time. A probabilistic polynomial-time algorithm can only perform a polynomial number of operations, including at most a polynomial number of coin flips.
The above calculation has the property of duality. The receiver can also generate his one-time address on his own. Specifically, the receiver takes as input two random numbers $r_1\prime, \rho\prime \in Z_n^*$, his long-time private key $sk_2 = (a_2, b_2)$, the first part of the long-time public key A_1 of the sender, and the first part of the long-time public key A_3 of the supervisor. The following computation is presented:

$$\begin{aligned} r_2\prime &:= \text{hash}_1(r_1\prime, a_2 \cdot A_1), \\ r_3\prime &:= \text{hash}_2(r_2\prime \cdot A_3), \\ y_1\prime &:= \text{hash}_2(r_3\prime \cdot G) \oplus \rho\prime, \\ R\prime &:= r_2\prime \cdot G, \\ \text{PK}_2^{\text{OneTime}}\prime &:= \rho\prime \cdot G + R\prime + B_2 \\ y_2 &:= \text{hash}_1(r_3\prime, y_1\prime, \rho\prime, B_2, \text{PK}_2^{\text{OneTime}}\prime) \end{aligned}$$

Let $\text{Addr} = (\text{PK}_2^{\text{OneTime}}\prime, R\prime, y_1\prime, y_2\prime, r_1\prime)$. Therefore, the sender and the receiver can generate the one-time address, but only the receiver can accurately generate the corresponding one-time private key. Additional discussion will be provided later.
- **OnetimeSKGen.** The receiver checks the structure of the one-time address

$$y_2 == \text{hash}_1(r_3, y_1, \rho, B_2, \text{PK}_2^{\text{OneTime}})$$

If incorrect, then reject; otherwise takes as input his long-time private key $\text{sk}_2 = (a_2, b_2)$, the one-time address $\text{Addr} = (\text{PK}_2^{\text{OneTime}}, R, y_1, y_2, r_1)$, the first part of the long-time public keys of the receiver A_1 and the supervisor A_3. Then, computes as follows:

$$r_2 := \text{hash}_1(r_1, a_2 \cdot A_1),$$
$$r_3 := \text{hash}_2(r_2 \cdot A_3),$$
$$\rho := \text{hash}_2(r_3 \cdot G) \oplus y_1$$
$$\text{sk}_2^{\text{OneTime}} := \rho + r_2 + b_2$$

The receiver accepts the transaction if the following two equations hold

$$R == r_2 \cdot G$$
$$\text{PK}_2^{\text{OneTime}} == \text{sk}_2^{\text{OneTime}} \cdot G$$

These two consistency checks ensure that the one-time address Addr was generated correctly. If the equation holds, then the one-time private key $\text{sk}_2^{\text{OneTime}}$ calculated by the receiver corresponds to the one-time public key $\text{PK}_2^{\text{OneTime}}$.

- **IdentityTracing.** The supervisor takes as input the first part of his long-time private key a_3 and the one-time address $\text{Addr} = (\text{PK}_2^{\text{OneTime}}, R, y_1, y_2, r_1)$.Then, computes as follows

$$r_3 := \text{hash}_2(a_3 \cdot R),$$
$$\rho := \text{hash}_2(r_3 \cdot G) \oplus y_1$$
$$B_2 := \text{PK}_2^{\text{OneTime}} - \rho \cdot G - R$$

Therefore, the supervisor can trace the long-time public key B_2 of the receiver by using the one-time address Addr and his long-time private key a_3.

- **SenderStatistics.** According to the trade of digital signature to find the one-time address $\text{Addr} = (\text{PK}_2^{\text{OneTime}}, R, y_1, y_2, r_1)$, the sender takes as input his private key a_1, and the long-term public key of the intended recipient, (a_2, B_2). The local long-term public key of the regulator A_3 is calculated as follows:

$$r_2 := \text{hash}_1(r_1, a_1 \cdot A_2),$$
$$r_3 := \text{hash}_2(r_2 \cdot A_3),$$
$$\rho := \text{hash}_2(r_3 \cdot G) \oplus y_1$$

Check if the following equation is holds

$$R == r_2 \cdot G$$
$$B_2 == \text{PK}_2^{\text{OneTime}} - \rho \cdot G - R$$

If yes, then accept, otherwise reject. Therefore, the sender can use his long-term private key a_1 and the one-time address Addr to verify whether the long-term public key B_2 of the target receiver is correct. Therefore, he can trace the identity of the receiver and realize the identity statistics of recipients.

- **ReceiverStatistics.** The receiver takes as input his long-time private key $a_2$and his long-time public key B_2, the long-time public key $a_1$of the sender, the long-time public key $a_3$of the supervision, and the one-time address $\text{Addr} = (\text{PK}_2^{\text{OneTime}}, R, y_1, y_2, r_1)$. The following computation is presented:

$$r_2 := \text{hash}_1(r_1, a_2 \cdot A_1),$$
$$r_3 := \text{hash}_2(r_2 \cdot A_3),$$
$$\rho := \text{hash}_2(r_3 \cdot G) \oplus y_1$$

Check if the following equation is holds

$$R == r_2 \cdot G$$
$$B_2 == \text{PK}_2^{\text{OneTime}} - \rho \cdot G - R$$

If yes, then accept; otherwise reject. Therefore, the receiver can use his long-time private key a_2 and one-time address Addr to verify whether the long-time public key of the sender is A_1. Therefore he can trace the identity of the sender and realize the identity statistics of senders.

Theorem 3. *Only the corresponding receiver can correctly compute the one-time private key according to the one-time public key generated by the sender.*

Proof 3. The random number $r_1$in one-time address Addr is public, and the sender calculation of two random numbers r_2, r_3as follows

$$r_2 := \text{hash}_1(r_1, a_1 \cdot A_2) = \text{hash}_1(r_1, a_1 a_2 \cdot G),$$
$$r_3 := \text{hash}_2(r_2 \cdot A_3)$$

By contrast, the two random numbers computed by the receiver are as follows $\tilde{r}_2, \tilde{r}_3$

$$\tilde{r}_2 := \text{hash}_1(r_1, a_2 \cdot A_1) = \text{hash}_1(r_1, a_2 a_1 \cdot G),$$
$$\tilde{r}_3 := \text{hash}_2(r_2 \cdot A_3)$$

Therefore, we have

$$r_2 == \tilde{r}_2$$
$$r_3 == \tilde{r}_3$$

Therefore, the sender and the receiver can calculate the random number r_2, r_3 from different angles. Thus, the receiver can correctly calculate as follows

$$\rho := \text{hash}_2(r_3 \cdot G) \oplus y_1$$

The one-time public key of the receiver is calculated by the sender as follows

$$\text{PK}_2^{\text{OneTime}} := \rho \cdot G + R + B_2$$

The corresponding one-time private key calculated by the receiver as follows

$$\text{sk}_2^{\text{OneTime}} := \rho + r_2 + b_2$$

The following discrete logarithm relationship holds

$$\text{sk}_2^{\text{OneTime}} \cdot G = (\rho + r_2 + b_2) \cdot G$$
$$= \rho \cdot G + R + B_2 = \text{PK}_2^{\text{OneTime}}$$

Therefore, the one-time private key $\text{sk}_2^{\text{OneTime}}$ calculated by the receiver has a discrete logarithmic relationship with the one-time public key $\text{PK}_2^{\text{OneTime}}$ generated by the sender.

Moreover, as the one-time private key $\text{sk}_2^{\text{OneTime}}$ is the sum of r_3, r_2 and b_2, although the sender and the supervisor compute r_3, they cannot compute the second part of the long-time private key of the receiver b_2. Therefore, only the receiver can compute $\text{sk}_2^{\text{OneTime}}$, which means the amount paid to the one-time address Addr can only be spent by the receiver. This completes the proof of Theorem 3. □

Algorithm 4: one-time key generation

Input: one-time private key $\text{sk}_2^{\text{OneTime}}$
Output: one-time public key generated by the sender
For each sender
  Compute random numbers r_2, r_3
  For each receiver
    Compute random number $\tilde{r}_2, \tilde{r}_3$
    If$(r_2 == \tilde{r}_2, r_3 == \tilde{r}_3)$
      Both $\rho := \text{hash}_2(r_3 \cdot G) \oplus y_1$
      one-time public key → sender
        $\text{PK}_2^{\text{OneTime}} := \rho \cdot G + R + B_2$
        one-time private key → receiver
        $\text{sk}_2^{\text{OneTime}} := \rho + r_2 + b_2$
    End if
  End for
End for

Theorem 4. *The supervisor can correctly trace the long-time public key, that is, the identity of the receiver from the one-time public key by using the first part of his long-time private key a_3.*

Proof 4. The expression of r_3 and R computed by the sender is as follows

$$r_3 = \text{hash}_2(r_2 \cdot A_3) = \text{hash}_2(r_2 a_3 \cdot G)$$

By contrast, since the auxiliary information R is publicly available in the one-time address Addr, the supervisor can also calculate the random number r_3 as follows

$$\tilde{r}_3 := \text{hash}_2(a_3 \cdot R) = \text{hash}_2(a_3 r_2 \cdot G)$$

The following equation holds $\tilde{r}_3 == r_3$
Therefore, the supervisor can calculate as follows

$$\rho := \text{hash}_2(r_3 \cdot G) \oplus y_1$$

Thus, the supervisor can compute the long-term public key of the receiver from the one-time address.

$$B_2 := \text{PK}_2^{\text{OneTime}} - \rho \cdot G - R$$

This completes the proof of Theorem 2. □

Algorithm 5: traceability of the sender

Input: long-time private key a_3
Output: identity the receiver from the one-time public key
For each transaction
  Evaluate $r_3 := \text{hash}_2(r_2 \cdot A_3) = \text{hash}_2(r_2 a_3 \cdot G)$
  Compute one-time address Addr
  $\tilde{r}_3 := \text{hash}_2(a_3 \cdot R) = \text{hash}_2(a_3 r_2 \cdot G)$
  If$(\tilde{r}_3 == r_3)$
    The long-time public key of the sender computed by the supervisor
    $\rho := \text{hash}_2(r_3 \cdot G) \oplus y_1$
    $B_2 := \text{PK}_2^{\text{OneTime}} - \rho \cdot G - R$
  End if
End for
Return: completes the proof

6. Security

6.1. Security model

Indistinguishability against chosen-plaintext attacks (IND-CPA) and indistinguishability against chosen-ciphertext attacks (IND-CCA) are two public-key encryption techniques (IND-CCA). Indistinguishability under chosen plaintext attack (IND-CPA) is characterized as a game between an adversary and a challenger for a probabilistic asymmetric key encryption scheme. An adversary in computing systems is a polynomial time Turing machine, which must finish the game and emit a "guess" in a polynomial number of steps. The indistinguishability definitions for non-adaptive and adaptive Chosen Ciphertext Attack (IND-CCA, IND-CCA2) are the same as for IND-CPA. In addition to the public key, the adversary has access to a "decryption oracle", which decrypts arbitrary ciphertexts and delivers the plaintext at the request of the adversary.

The security model of the traceable one-time address scheme is defined. Without the secret key sk, extracting the long-time public key PK from the given one-time Address Addr $= (\text{PK}^{\text{OneTime}}, R, r_1)$ is difficult for any probabilistic polynomial-time adversary.

The indistinguishability security of the traceable one-time address scheme is modeled by a game played by a challenger and an adversary. The challenger generates a traceable one-time address scheme, while the adversary attempts to break the scheme. First, the challenger generates three key pairs $(\text{PK}_A, \text{sk}_A), (\text{PK}_B, \text{sk}_B), (\text{PK}_C, \text{sk}_C)$, and sends the long-time public keys $\text{PK}_A, \text{PK}_B, \text{PK}_C$to the adversary, and keeps the secret keys $\text{sk}_A, \text{sk}_B, \text{sk}_C$. The adversary outputs two distinct long-time public keys PK_0, PK_1 from the same public key space to be challenged. The challenger generates a challenge one-time address Addr^* on one of the long-time public key PK_ξrandomly chosen from $\{\text{PK}_0, \text{PK}_1\}$. If tracing queries are allowed, then the adversary can make tracing queries on any one-time address that is adaptively chosen by the adversary itself with the restriction that no tracing query is allowed on challenge one-time address Addr^*. Finally, the adversary outputs a guess of the chosen one-time public key PK_ξ in the challenge one-time address Addr^*.

Formally, the security model of the indistinguishability public key against IND-PK-CCA can be described as follows.

- **Setup.** Let SP be the system parameters. The challenger runs the key generation algorithm to generate a key pair (PK, sk) and sends PK to the adversary. The challenger keeps sk to respond to tracing queries from the adversary.

- **Phase 1.** The adversary makes tracing queries on one-time addresses that are adaptively chosen by the adversary itself. For a tracing query on the one-time address $\text{Addr}_i, 1 \leqslant i \leqslant q_T$, the challenger runs the tracing algorithm and then sends the tracing result to the adversary.
- **Challenge.** The adversary outputs two distinct long-time public keys PK_0, PK_1 from the same public key space, which is adaptively chosen by the adversary itself. The challenger randomly chooses a bit $\xi, \xi \in \{0, 1\}$ and then computes a challenge one-time address Addr^*, which is given to the adversary.
- **Phase 2.** The challenger responds to tracing queries in the same way as in Phase 1 with the restriction that no tracing query is allowed on Addr^*.
- **Guess.** The adversary outputs a guess $\xi\prime$ of ξ and wins the game if $\xi = \xi\prime$. The advantage $\varepsilon$of the adversary in winning this game is defined as

$$\varepsilon := 2\left(\Pr[\xi == \xi\prime] - \frac{1}{2}\right)$$

Definition 1. (IND-PK-CCA) A traceable one-time address scheme is (t, q_T, ε)-secure in the IND-PK-CCA security model in the absence of an adversary who can win the above game in time t with advantage ε after making q_T tracing queries. IND-CCA is widely recognized for many cryptographic applications. However, IND-CPA security, that is, indistinguishability (and one-way) against selected plaintext attacks, is typically significantly difficult to verify.

The IND-PK-CCA model is the standard security model for the traceable one-time address scheme. A weak version of the indistinguishability security model, namely indistinguishability public key against chosen-plaintext attacks (IND-PK-CPA), is also referred to as semantic security and defined as follows.

Definition 2. (IND-PK-CPA) A traceable one-time address scheme is (t, ε)-secure in the security model of indistinguishability public key against chosen-plaintext attacks (IND-PK-CPA) if the scheme is $(t, 0, \varepsilon)$-secure in the IND-PK-CCA security model, where the adversary is not allowed to make any tracing query.

Theorem 5. *Suppose the hash function* hash_1 *is a random oracle. If the CDH problem is difficult, then the traceable one-time address scheme is provably secure in the IND-PK-CPA security model with reduction loss* $L = q_{\text{hash}_1}$*, where* q_{hash_1} *is the number of hash queries to the random oracle.*

Theorem 6. *Suppose the hash function* hash_1 *is a random oracle. If the CDH problem is difficult, then the traceable one-time address scheme is provably secure in the IND-PK-CPA security model with reduction loss* $L = q_{\text{hash}_1}$*, where* q_{hash_1} *is the number of hash queries to the random oracle.*

The first instance has IND-PK-CPA security in this above security model. Moreover, the REACT conversion introduced by Okamoto and Pointcheval [40] can convert an IND-CPA secure scheme into an IND-CCA secure scheme. So our second instance has IND-PK-CCA security.

Proof 5. Suppose there exists an adversary $\mathscr{A}$, who can (t, ε)-break the traceable one-time address scheme in the above IND-PK-CPA security model. Then, we can construct a simulator $\mathscr{B}$ to solve the CDH problem. The computational Diffie-Hellman (CDH) assumption states that a given computing issue in a cyclic group is difficult. The cyclic group G, which has the order q, is considered. Given a randomly generated generator g and random, the CDH assumption asserts that computing the value is computationally intractable. Given as input a problem instance $(G, \alpha \cdot G, \beta \cdot G)$ over the cyclic group $(\mathbb{G}, G, n)$, the simulator $\mathscr{B}$ runs the adversary $\mathscr{A}$ and works as follows.

- **Setup.** Let $\text{SP} = (\mathbb{G}, G, n)$and $\text{hash}_1, \text{hash}_2$ be the two random oracles controlled by the simulator $\mathscr{B}$. The simulator $\mathscr{B}$ sets the long-time public key as $\text{PK}_A = \alpha \cdot G$ where $\text{sk}_A = \alpha$. The long-time public key PK_A is one of the problem instance $\alpha \cdot G$.
- **H-Query.** The adversary $\mathscr{A}$ makes hash queries in this phase. The simulator $\mathscr{B}$ prepares two hash lists to record all queries and responses as follows. In the beginning, the hash list is empty.
  Let the i-th hash query be (x_i, G_i). If (x_i, G_i) is already in the hash list, then the simulator $\mathscr{B}$ responds to this query following the hash list. Otherwise, the simulator $\mathscr{B}$ randomly chooses $y_i \in Z_n$ and sets $y_i = \text{hash}_1(x_i, G_i)$. The simulator $\mathscr{B}$ responds to this query with $\text{hash}_1(x_i, G_i)$ and adds (x_i, G_i, y_i) to the hash list.
- **Challenge.** The adversary $\mathscr{A}$ outputs two distinct long-time public keys PK_0, PK_1 to be challenged, and a random number r_1^*. The simulator $\mathscr{B}$ randomly chooses a group element ψ^*, and computes as follows

$$\begin{aligned} r_2^* &:= \text{hash}_1(r_1^*, \psi^*) \\ r_3^* &:= \text{hash}_2(r_2^* \cdot \text{PK}_C) \\ R^* &:= r_2^* \cdot G \\ \text{PK}^{\text{OneTime}*} &:= r_3 \cdot G + R^* + \text{PK}_\xi \end{aligned}$$

The simulator $\mathscr{B}$ sets the challenge one-time address as

$$Addr^* = (\mathrm{PK}^{\mathrm{OneTime}*}, R^*, r_1^*).$$

The challenge one-time address can be regarded as a generation of the message $\mathrm{PK}_\xi \in \{\mathrm{PK}_0, \mathrm{PK}_1\}$ using the random coin ξ, if $\psi^* = \beta \cdot (\alpha \cdot G)$

$$\begin{aligned} r_2^* &= \mathrm{hash}_1(r_1^*, \psi^*) = \mathrm{hash}_1(r_1^*, \alpha\beta \cdot G) \\ r_3^* &= \mathrm{hash}_2(r_2^* \cdot \mathrm{PK}_C) \\ R^* &= r_2^* \cdot G \\ \mathrm{PK}^{\mathrm{OneTime}*} &= r_3 \cdot G + R^* + \mathrm{PK}_\xi \end{aligned}$$

Let $Addr^* = (\mathrm{PK}^{\mathrm{OneTime}*}, R^*, r_1^*)$ be the challenge one-time address, which is a correct one-time address from the point of view of the adversary, in the absence of hash query on $\beta \cdot \alpha G$ to the random oracle.

- **Guess.** The adversary $\mathscr{A}$ outputs a guess or $\perp$. The challenge hash query is defined as

$$Q^* = \psi^* = \alpha\beta \cdot G$$

The simulator $\mathscr{B}$ randomly selects one value G_i from the hash list $G_1, G_2, \ldots, G_{q_{\mathrm{hash}_1}}$ as the challenge hash query. The simulator can immediately use this hash query to solve the CDH problem.

This completes the simulation and the solution. The correctness is analyzed as follows.

- **Indistinguishable simulation.** The correctness of the simulation has been explained above. The randomness of the simulation includes all random numbers in the key generation, the responses to hash queries, and the challenge one-time address. They are

$$\alpha, y_1, \ldots, y_{q_{\mathrm{hash}_1}}, \beta$$

According to the setting of the simulation, where α, β, y_i are randomly chosen, it is easy to see that the randomness property holds, and thus the simulation is indistinguishable from the real attack.

- **Probability of successful simulation.** No abort is found in the simulation; thus, the probability of successful simulation is 1.
- **Advantage of breaking the challenge one-time address.** The challenge one-time address is a generation of PK_0, if

$$\begin{aligned} r_2^* &= \mathrm{hash}_1(r_1^*, \psi) = \mathrm{hash}_1(r_1^*, \alpha\beta \cdot G) \\ r_3^* &= \mathrm{hash}_2(r_2^* \cdot \mathrm{PK}_0) \\ R^* &= r_2^* \cdot G \\ PK^{\mathrm{OneTime}*\,\mathrm{OneTime}*} &= r_3 \cdot G + R^* + \mathrm{PK}_0 \end{aligned}$$

The challenge one-time address is a generation of PK_1, if

$$\begin{aligned} r_2^* &= \mathrm{hash}_1(r_1^*, \psi) = \mathrm{hash}_1(r_1^*, \alpha\beta \cdot G) \\ r_3^* &= \mathrm{hash}_2(r_2^* \cdot \mathrm{PK}_1) \\ R^* &= r_2^* \cdot G \\ PK^{\mathrm{OneTime}*\,\mathrm{OneTime}*} &= r_3 \cdot G + R^* + \mathrm{PK}_1 \end{aligned}$$

If the query $\alpha\beta \cdot G$ is not made, then

$$\mathrm{hash}_2\big(\mathrm{hash}_1(r_1^*, \alpha\beta \cdot G) \cdot PK_C\big)$$

is random and unknown to the adversary, so he has no advantage in breaking the challenge one-time address.

- **Probability of finding a solution.** The adversary has an advantage εin guessing the chosen message according to the breaking assumption, the adversary will query $\alpha\beta \cdot G$ to the random oracle with probability ε. Thus, the adversary makes q_{hash_1} hash queries in total. Therefore, a random choice of G_i is equal to $\alpha\beta \cdot G$ with probability $\varepsilon/q_{\mathrm{hash}_1}$.
- **Advantage and time cost.** Let T_s denote the time cost of the simulation; Thus, $T_s = O(1)$. Therefore, the simulator $\mathscr{B}$ will solve the CDH problem in time $t + T_s$ with advantage $\varepsilon/q_{\mathrm{hash}_1}$.

This completes the proof of the Theorem 5. □

7. Comparison

Table 1 shows the comparison of the proposed scheme with CryptoNote V2.0. In Table 1, from the second to the fourth column, demonstrate the length of the long-time private key |sk|, long-time public key |PK|, and the length of the one-time

Table 1
Functions and Computational Complexity.

	|sk|	|PK|	|Addr|	AddrGen	$sk^{OneTime}Gen$	Trace
CryptoNote V2.0	$2n$	$2\|G\|$	$2\|G\|$	1hash + 3exp	1hash + 1exp	–
First instance	$2n$	$2\|G\|$	$n+2\|G\|$	2hash + 4exp	2hash + 2exp	1hash + 1exp
Second instance	$2n$	$2\|G\|$	$3n+2\|G\|$	3hash + 5exp	3hash + 2exp	3hash + 3exp

Table 2
Experimental Test.

	AddrGen (ms)	$sk^{OneTime}$Gen (ms)	Trace (ms)	SenderStatistics (ms)	ReceiverStatistics (ms)
CryptoNote V2.0	1.67	0.56	–	–	–
First instance	2.23	1.11	0.56	1.13	1.12
Second instance	2.56	1.20	0.82	1.57	1.54

address |Addr|. From the fifth to the seventh column, it demonstrates the computation complexity of the one-time address generation algorithm AddrGen, the one-time private key generation algorithm $sk^{OneTime}Gen$, and the tracing algorithm *Trace*, respectively. In Table 1, hash denote the hash computational complexity, and *Exp* denotes the exponential computational complexity.

The table shows that the proposed scheme has the same length of the long-time private and public keys with CryptoNote V2.0. The length of the introduced one-time address to achieve traceability is only n bits longer than the CryptoNote V2.0; the computation complexity of the first instance of the one-time address generation/one-time private key generation is only 1hash + 1exp/1hash + 1exp more complex than the CryptoNote V2.0, respectively. the computation complexity of the second instance of the one-time address generation/one-time private key generation is only 2hash + 2exp/2hash + 1exp more complex than the CryptoNote V2.0, respectively.

8. Experimental results

A traceable one-time address scheme and the one-time address scheme of CryptoNote V2.0 are implemented on the secp256k1 curve in C programming language, which contains approximately 300/200 lines of code, respectively. The Bitcoin system is implemented using the Secp256k1 elliptic curve. Every point on this graph represents a valid Bitcoin public key. When a user wants to produce a public key from their private key, they multiply their private key by a large number and then divide the result by the Generator Point, which is a specified point on the secp256k1 curve.

In the implementation, the hash function is SHA-256. The computer information: Keynel x86_64 Linux 4.17.6–1-ARCH with 1 core Intel i5-8250U 1.60 GHz and 16 GB RAM. Experiments show that the speed of our scheme and the one-time address scheme of CryptoNote V2.0 depends on the concrete data. Therefore, multiple data tests are performed. Experimental results show that the average speed of address generation algorithms of CryptoNote V2.0 and the two instances are $1.67/2.23/2.56ms$, respectively; the average speed of one-time private key generation algorithms of CryptoNote V2.0 and the two instances is $0.56/1.11/1.20ms$, respectively. The average speed of the two tracing algorithms is $0.56/0.82ms$. The average speed of the two *SenderStatistics* algorithms is $1.13/1.57ms$, while that of the two *ReceiverStatistics* algorithms is $1.12/1.54ms$. The speed of the proposed scheme and CryptoNote V2.0 is shown in Table 2.

9. Conclusion

Therefore, we proposed a traceable one-time address scheme to provide both anonymity for receivers and traceability for the system supervisor. In the scheme, the sender can calculate a one-time address using the receiver's public key or address. Then, he can pay some amount to the one-time address. However, only the right receiver can calculate the corresponding one-time private key by using his long-time private key. Since the one-time address is used only once, the recipient is anonymous. However, the supervisor of the system can compute the long-time public key from the one-time address. Thus, the supervisor can trace the identity of the anonymous user. We defined two security models, constructed two scheme instances, and proved that the two instances are secure in the IND-PK-CPA/IND-PK-CCA model, respectively. We implement our two instances and the one-time generation scheme of CryptoNote V2.0 in the C programming language. The one-time address generation time of our instances is $2.23/2.56ms$; the one-time private key generation time is $1.11/1.20ms$; the trace time is $0.56/0.82ms$. Although our scheme is slower than CryptoNote 2.0, we added 3 functions to the scheme, namely, traceability, sender statistics, and receiver statistics. Finally, we build a blockchain system for Digital Museum assets and implement our two instances in the system.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Jiawan Zhang reports administrative support and writing assistance were provided by Tianjin University.

Acknowledgements

This work was supported by the study and demonstration application of organization and service of cultural heritage knowledge graph, National Key Research and Development Program of China (Grant No. 2019YFC1521200), and Large scale literature data visual analysis based on sciencemap, National Natural Science Foundation of China (Grant No. 621 72295).

References

[1] R. Ali, J. Barrdear, R. Clews, J. Southgate, The economics of digital currencies, Bank of Engl. Q. Bull. Q3 (2014).
[2] E. Bertacchini, F. Morando, The future of museums in the digital age: New models for access to and use of digital collections, Int. J. Arts Manage. 15 (2) (2013) 60–72.
[3] N. Andola, V.K. Yadav, S. Venkatesan, S. Verma, et al, Anonymity on blockchain based e-cash protocols-A survey, Comput. Sci. Rev. 40 (2021) 100394.
[4] M.C.K. Khalilov, A. Levi, A survey on anonymity and privacy in bitcoin-like digital cash systems, IEEE Commun. Surveys Tutor. 20 (3) (2018) 2543–2585.
[5] E. Androulaki, G.O. Karame, M. Roeschlin, T. Scherer, S. Capkun, Evaluating user privacy in bitcoin, in: International conference on financial cryptography and data security, Springer, 2013, pp. 34–51.
[6] J.V. Monaco, Identifying bitcoin users by transaction behavior, in: Biometric and surveillance technology for human and activity identification XII, vol. 9457, SPIE, 2015, pp. 25–39.
[7] O. Marian, A conceptual framework for the regulation of cryptocurrencies, U. Chi. L. Rev. Dialogue 82 (2015) 53.
[8] R. Goyat, G. Kumar, M. Alazab, M. Conti, M.K. Rai, R. Thomas, R. Saha, T.-H. Kim, Blockchain-Based Data Storage With Privacy and Authentication in Internet of Things, IEEE Internet Things J. 9 (16) (2020) 14203–14215.
[9] N. Van Saberhagen, CryptoNote v 2.0, 2013.
[10] F. Tschorsch, B. Scheuermann, Bitcoin and beyond: A technical survey on decentralized digital currencies, IEEE Commun. Surveys Tutor. 18 (3) (2016) 2084–2123.
[11] H.H. Sun Yin, K. Langenheldt, M. Harlev, R.R. Mukkamala, R. Vatrapu, Regulating cryptocurrencies: a supervised machine learning approach to de-anonymizing the bitcoin blockchain, J. Manage. Inf. Syst. 36 (1) (2019) 37–73.
[12] E. Rehman, M.A. Khan, T.R. Soomro, N. Taleb, M.A. Afifi, T.M. Ghazal, Using blockchain to ensure trust between donor agencies and ngos in under-developed countries, Computers 10 (8) (2021) 98.
[13] J. Groth, On the size of pairing-based non-interactive arguments, in: Annual international conference on the theory and applications of cryptographic techniques, Springer, 2016, pp. 305–326.
[14] S. Bowe, A. Gabizon, M.D. Green, A multi-party protocol for constructing the public parameters of the Pinocchio zk-SNARK, in: International Conference on Financial Cryptography and Data Security, Springer, 2018, pp. 64–77.
[15] E. Ben-Sasson, I. Bentov, Y. Horesh, M. Riabzev, Scalable, transparent, and post-quantum secure computational integrity, Cryptology ePrint Archive (2018).
[16] S. Bowe, J. Grigg, D. Hopwood, Recursive proof composition without a trusted setup, Cryptology ePrint Archive (2019).
[17] A. Gabizon, Z.J. Williamson, O. Ciobotaru, Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge, Cryptology ePrint Archive (2019).
[18] B. Chen, B. Bünz, D. Boneh, Z. Zhang, HyperPlonk: Plonk with Linear-Time Prover and High-Degree Custom Gates, Cryptology ePrint Archive (2022).
[19] I. Miers, C. Garman, M. Green, A.D. Rubin, Zerocoin: Anonymous distributed e-cash from bitcoin, in: 2013 IEEE Symposium on Security and Privacy, IEEE, 2013, pp. 397–411.
[20] E.B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, M. Virza, Zerocash: Decentralized anonymous payments from bitcoin, in: 2014 IEEE Symposium on Security and Privacy, IEEE, 2014, pp. 459–474.
[21] S. Bowe, A. Chiesa, M. Green, I. Miers, P. Mishra, H. Wu, Zexe: Enabling decentralized private computation, in: 2020 IEEE Symposium on Security and Privacy (SP), IEEE, 2020, pp. 947–964.
[22] P. Daian, S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, A. Juels, Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability, in: 2020 IEEE Symposium on Security and Privacy (SP), IEEE, 2020, pp. 910–927.
[23] T. Kerber, A. Kiayias, M. Kohlweiss, Kachina–foundations of private smart contracts, in: 2021 IEEE 34th Computer Security Foundations Symposium (CSF), IEEE, 2021, pp. 1–16.
[24] F. Engelmann, L. Müller, A. Peter, F. Kargl, C. Bösch, SwapCT: Swap confidential transactions for privacy-preserving multi-token exchanges, Cryptology ePrint Archive (2021).
[25] N. Glaeser, M. Maffei, G. Malavolta, P. Moreno-Sanchez, E. Tairi, S.A.K. Thyagarajan, Foundations of coin mixing services, in: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, 2022, pp. 1259–1273.
[26] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J.A. Kroll, E.W. Felten, Mixcoin: Anonymity for bitcoin with accountable mixes, in: International Conference on Financial Cryptography and Data Security Springer, 2014, pp. 486–504.
[27] L. Valenta, B. Rowan, Blindcoin: Blinded, accountable mixes for bitcoin, in: International Conference on Financial Cryptography and Data Security Springer, 2015, pp. 112–126.
[28] J.H. Ziegeldorf, F. Grossmann, M. Henze, N. Inden, K. Wehrle, Coinparty: Secure multi-party mixing of bitcoins, in: Proceedings of the 5th ACM Conference on Data and Application Security and Privacy, 2015, pp. 75–86.
[29] G. Maxwell, CoinJoin: Bitcoin privacy for the real world, Post on Bitcoin forum, 2013.
[30] T. Ruffing, P. Moreno-Sanchez, A. Kate, Coinshuffle: Practical decentralized coin mixing for bitcoin, in: European Symposium on Research in Computer Security Springer, 2014, pp. 345–364.
[31] S. Meiklejohn, C. Orlandi, Privacy-enhancing overlays in bitcoin, in: International Conference on Financial Cryptography and Data Security Springer, 2015, pp. 127–141.
[32] M.F. Esgin, R. Steinfeld, A. Sakzad, J.K. Liu, D. Liu, Short lattice-based one-out-of-many proofs and applications to ring signatures, in: International Conference on Applied Cryptography and Network Security Springer, 2019, pp. 67–88.
[33] M. Backes, N. Döttling, L. Hanzlik, K. Kluczniak, J. Schneider, Ring signatures: logarithmic-size, no setup–from standard assumptions, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques Springer, 2019, pp. 281–311.
[34] W.A. Alberto Torres, R. Steinfeld, A. Sakzad, J.K. Liu, V. Kuchta, N. Bhattacharjee, M.H. Au, J. Cheng, Post-quantum one-time linkable ring signature and application to ring confidential transactions in blockchain (lattice RingCT v1. 0), in: Australasian Conference on Information Security and Privacy Springer, 2018, pp. 558–576.
[35] S. Noether, Ring signgnature Confidential Transactions for Monero, IACR Cryptol. ePrint Arch 1098 (2015).

[36] T.H. Yuen, S. Sun, J.K. Liu, M.H. Au, M.F. Esgin, Q. Zhang, D. Gu, RingCT 3.0 for Blockchain Confidential Transaction: Shorter Size and Stronger, Security 12059 (2020) 464–483.
[37] S.-F. Sun, M.H. Au, J.K. Liu, T.H. Yuen, Ringct 2.0: A compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero, in: European Symposium on Research in Computer Security Springer, 2017, pp. 456–474.
[38] X. Lu, M.H. Au, Z. Zhang, Raptor: a practical lattice-based (linkable) ring signature, in: International Conference on Applied Cryptography and Network Security Springer, 2019, pp. 110–130.
[39] E. Fujisaki, K. Suzuki, Traceable ring signature, in: International Workshop on Public Key Cryptography, Springer, 2007, pp. 181–200.
[40] T. Okamoto, D. Pointcheval, REACT Rapid enhanced-security asymmetric cryptosystem transform, in: Cryptographers' Track at the RSA Conference, Springer, 2001, pp. 159–174.