

Article

An Origin-Traceable Address Derivation Scheme with Provable Linkage for Blockchain Applications

Jianing Tan, Chao Xu * and Fanfan Shen

School of Computer Science, Nanjing Audit University, Nanjing 211815, China; jn_tan@163.com (J.T.); ffshen@nau.edu.cn (F.S.)

* Correspondence: xuchao@nau.edu.cn; Tel.: +86-13770771835

Abstract

A blockchain address serves as the unique identifier for users within a blockchain network. While existing blockchain solutions typically employ the Elliptic Curve Digital Signature Algorithm (ECDSA) for address generation, researchers including Ruimin Wang and Jisheng Dong have proposed alternative schemes based on SM9 identity-based cryptography to circumvent the inherent limitations of the Public Key Infrastructure (PKI). However, while preserving the address privacy of transacting parties, these schemes lack traceability, creating vulnerabilities to Sybil attacks in application scenarios such as voting. To address this issue, this paper proposes a novel blockchain address generation scheme based on SM9 identity-based cryptography. Incorporating the concept of hierarchical cryptography, we design a two-tier key derivation structure to guarantee the address privacy of both transacting parties. Furthermore, the generation of derived keys integrates secret sharing technology, thereby achieving source traceability with provable linkage. Provided that participants have registered a legitimate identity (blockchain address), utilizing the proposed scheme to construct derived addresses effectively prevents them from launching Sybil attacks via these derived addresses. The security properties of the proposed scheme—including uniqueness, unpredictability, pseudonymity, traceability, and Sybil resistance—are formally proven under the random oracle model. Finally, theoretical analysis and experimental results demonstrate that, alongside enhanced security (100% Source Tracing Accuracy), the proposed scheme incurs no significant increase in key and address lengths compared to existing solutions, with only a marginal increase in computational overhead.

Keywords: blockchain address; the SM9 algorithm; secret sharing; the ECDSA algorithm



Academic Editor: Hung-Yu Chien

Received: 8 May 2026

Revised: 29 June 2026

Accepted: 3 July 2026

Published: 9 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Blockchain technology, originating from Bitcoin [1], enables the establishment of peer-to-peer trusted connections among mutually untrusted nodes without a central authority, thereby facilitating secure data exchange in distributed, untrusted environments. Unlike traditional data management systems, blockchain operates as a distributed shared ledger underpinned by a chain-like data structure [2]. All records, authenticated via digital signatures, are stored in blocks that are cryptographically linked to form the chain. Characterized by decentralization, data immutability, openness, pseudonymity, robust security, and transparency, blockchain has been widely adopted across diverse domains, including finance, supply chain management, and the Internet of Things (IoT). This widespread adoption underscores its substantial practical utility and significant potential for future development [3–5].

A blockchain address functions as a user's unique digital identity within the network, serving as a public identifier for sending and receiving digital assets such as cryptocurrencies. In traditional blockchain systems, such as Bitcoin and Ethereum, user addresses are predominantly derived from key pairs generated via elliptic curve cryptography (ECC), inherently relying on Public Key Infrastructure (PKI). However, this PKI-dependent approach introduces significant challenges, including cumbersome certificate management, complex deployment, and reduced operational efficiency. Identity-Based Cryptography (IBC) [6] mitigates these issues by allowing an arbitrary string that uniquely identifies a user to serve directly as their public key, thereby obviating the need for digital certificates and complex PKI frameworks. Consequently, the integration of blockchain technology with IBC has garnered significant research attention. For instance, the authors in [7] utilized identity strings as public keys to streamline key management, while Kaaniche and Laurent [8] proposed a user-centric data auditing scheme leveraging identity-based encryption (IBE) and blockchain. Furthermore, Momeni et al. [9] combined multi-party computation, homomorphic encryption, and IBE to implement a decentralized, leaderless sealed-bid auction mechanism.

While blockchain addresses are typically decoupled from users' real-world identities—thereby providing a degree of pseudonymity—the inherent transparency of public ledgers poses significant privacy risks. Attackers can de-anonymize users by aggregating and analyzing on-chain behavioral data [10]. To mitigate these vulnerabilities, cryptographic primitives such as ring signatures [11] have been introduced. Ring signatures allow any member of a designated group (the “ring”) to sign a message on behalf of the group, ensuring that a verifier can only confirm the signer's membership without revealing their specific identity. The CryptoNote protocol [12], proposed in 2013, pioneered the application of ring signatures to preserve transaction anonymity in blockchain systems, and subsequent studies [13,14] have further refined this approach. Alternatively, zero-knowledge proofs (ZKPs) offer a robust mechanism for privacy preservation by enabling a prover to validate a statement's truth without disclosing the underlying sensitive data. Various blockchain protocols [15–17] have leveraged ZKPs to cryptographically conceal user identities and transaction details, thereby achieving strong anonymity guarantees.

The SM9 algorithm, an identity-based asymmetric cryptographic standard adopted by ISO/IEC [18], has been utilized in various blockchain applications. For instance, Wang et al. [19] designed a blockchain address generation scheme based on an enhanced SM9 algorithm; however, it fails to support anonymous transactions. Similarly, Dong et al. [20] proposed a fixed-length linkable ring signature scheme using SM9 to preserve recipient address privacy. Nevertheless, this approach lacks the ability to verify whether derived addresses originate from the same source, leaving the system vulnerable to Sybil attacks in scenarios such as e-voting. In the context of securing on-chain user behavior, several studies [21–23] have developed trusted data management systems to ensure tamper resistance and privacy. However, the trust assumptions in [21,22] rely heavily on specific hardware vendors, increasing procurement and operational costs, while the approach in [23] is restricted to non-decentralized scenarios. To mitigate Sybil attacks—where adversaries generate numerous low-cost fake identities to compromise network security and fairness—researchers have proposed various consensus-level countermeasures [24–26]. Despite their effectiveness, these modifications inevitably increase communication overhead. Furthermore, the committee-based mechanism proposed in [24] compromises decentralization and introduces potential single points of failure. Alternatively, purely cryptographic approaches have been explored to resist Sybil attacks without altering the underlying consensus protocol. These include Proof-of-Stake (PoS)-based mechanisms [27], which elevate attack costs via staking deposits, and decentralized identity schemes [28], which

aggregate user identifiers. However, both categories frequently employ zero-knowledge proofs (ZKPs), which inherently suffer from computational efficiency bottlenecks [29]. More importantly, all the aforementioned studies primarily focus on establishing verifiable mechanisms during the initial identity registration phase. They largely overlook a critical vulnerability: preventing malicious participants from exploiting derived addresses to launch Sybil attacks after they have already registered a legitimate primary blockchain address.

This paper proposes a provably linkable and source-traceable address derivation scheme for blockchain applications. This scheme enables address tracing while building upon existing mechanisms that protect the address privacy of both transacting parties. Specifically, it addresses the scenario where users with registered legitimate identities generate derived addresses, effectively preventing malicious actors from exploiting these derived addresses to launch Sybil attacks. The main contributions of this paper are summarized as follows:

1. **System Model and Security Definitions:** We formalize the system model and algorithmic framework for a traceable blockchain address derivation scheme based on SM9 identity-based cryptography. We define a comprehensive security model that guarantees address privacy and pseudonymity for both transacting parties while enabling verifiable source linkage. This model captures essential security properties, including unpredictability, pseudonymity, and source traceability.
2. **Concrete Construction and Security Proofs:** We construct a concrete traceable address derivation scheme based on SM9. By incorporating the paradigm of hierarchical cryptography, the scheme allows users to autonomously derive addresses, thereby preserving transaction privacy. Furthermore, we integrate secret sharing techniques into the key derivation process to ensure the source traceability of the derived addresses. We formally prove the security of our construction under the random oracle model, demonstrating that it strictly satisfies all the security properties defined in our model.
3. **Performance Evaluation and Comparative Analysis:** We conduct comprehensive theoretical analysis and experimental evaluations of the proposed scheme. The results demonstrate that the addresses generated by our scheme are comparable in length to those of existing state-of-the-art schemes [20], incurring only a marginal increase in computational overhead. In exchange for this slight performance trade-off, our scheme uniquely supports the traceability of derived addresses (100% Source Tracing Accuracy) and provides robust protection against Sybil attacks, significantly expanding its applicability to a broader range of security-critical blockchain scenarios.

2. Preliminaries

This section introduces the fundamental concepts underlying our proposed scheme, including blockchain addresses, bilinear maps, the SM9 identity-based encryption algorithm, and secret sharing techniques.

2.1. Blockchain Address

Blockchain is a decentralized, tamper-resistant distributed ledger technology that ensures data integrity through cryptography and facilitates multi-party collaboration via consensus mechanisms. Within this network, a blockchain address functions as a user's unique identifier, acting as the destination for sending and receiving digital assets (e.g., cryptocurrencies) and interacting with smart contracts. Across mainstream blockchain platforms such as Bitcoin, Ethereum, Solana, and Binance Smart Chain (BSC), the specific implementations of address generation vary depending on the chosen asymmetric cryp-

tographic algorithms and encoding standards. Nevertheless, the core derivation process universally follows a unidirectional sequence: Private Key $\rightarrow$ Public Key $\rightarrow$ Address. The fundamental cryptographic principle dictates that while a private key uniquely determines its corresponding address, it is computationally infeasible to reverse-engineer the private key from the address. Most prominent blockchains, including Bitcoin and Ethereum-based ecosystems, employ the Elliptic Curve Digital Signature Algorithm (ECDSA) for this key generation process.

As the primary identifier for user identity and asset ownership, a blockchain address must exhibit the following essential properties to guarantee security, privacy, and systemic functionality:

1. **Uniqueness:** Each generated address must be globally unique within the network to prevent collision attacks and asset misrouting.
2. **Unpredictability:** The address generation mechanism must be cryptographically secure, ensuring it is computationally infeasible for adversaries to guess, enumerate, or pre-compute valid addresses.
3. **Pseudonymity:** A blockchain address does not inherently reveal the user's real-world identity. However, unlike strict anonymity, pseudonymity implies that a user's real identity could potentially be de-anonymized if the address is linked with auxiliary off-chain or on-chain behavioral data.

2.2. Bilinear Maps

Bilinear pairings (or bilinear maps) serve as a fundamental cryptographic primitive, providing the theoretical foundation for advanced cryptographic schemes such as aggregate signatures and identity-based encryption (IBE). Let G_1 and G_2 be additive cyclic groups, and G_T be a multiplicative cyclic group, all of the same prime order p . A bilinear map $e : G_1 \times G_2 \rightarrow G_T$ is defined as a mapping that satisfies the following properties: $e : G_1 \times G_2 \rightarrow G_T$, which satisfies the following properties:

1. **Bilinearity:** For all $P \in G_1$, $Q \in G_2$ and $a, b \in \mathbb{Z}_p$, there is $e(aP, bQ) = e(P, Q)^{ab}$.
2. **Non-degeneracy:** There exist $P \in G_1$ and $Q \in G_2$ such that $e(P, Q) \neq 1$.
3. **Efficient Computability:** There exists an efficient algorithm to compute $e(P, Q)$ for any P, Q .

2.3. SM9 Algorithm

SM9 leverages bilinear pairings on elliptic curves to facilitate secure encryption and digital signatures, thereby obviating the need for digital certificates. The SM9 standard comprises four primary components: the Digital Signature Algorithm, the Key Exchange Protocol, the Key Encapsulation Mechanism, and the Public Key Encryption Algorithm. The scheme proposed in this paper primarily builds upon the SM9 Digital Signature Algorithm. Let λ denote the security parameter, and let N be a large prime determined by λ . Let $\mathcal{BP} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, N)$ be an asymmetric bilinear group of prime order N associated with λ . Let $P_1 \in \mathbb{G}_1$ and $P_2 \in \mathbb{G}_2$ be the respective generators of the groups $\mathbb{G}_1$ and $\mathbb{G}_2$. Furthermore, let H_1 and H_2 be cryptographic hash functions mapping from $\{0, 1\}^*$ to $\mathbb{Z}_N^*$. According to the SM9 standard [18], the Digital Signature Algorithm comprises the following four polynomial-time algorithms:

- **System Setup:** The Key Generation Center (KGC) selects a random number $msk = \alpha \in \mathbb{Z}_N^*$ as the master signing private key, then computes the master public key as $p_{pub} = \alpha P_2 \in \mathbb{G}_2$. The public parameter g is set as $g = e(P_1, P_{pub})$.
- **User Key Generation:** The private key corresponding to user identity ID is computed as $sk_{ID} = (\frac{\alpha}{\alpha + H_1(ID) \parallel hid, N}) \bmod N \cdot P_1$ where hid is a private key generation function identifier.

- **Signature Generation:** Let M be the message to be signed. The signer selects a random number $r \in \mathbb{Z}_N^*$, then computes $w = g^r$, $h = H_2(M||w, N)$, $l = (r - h) \bmod N$ and $S = l \cdot sk_{ID}$, outputs (h, S) as the signature.
- **Signature Verification:** Upon receiving message M and signature (h, S) , the verifier computes $P = H_1(ID||hid, N) \cdot P_2 + P_{pub}$, $u = e(S, P)$ and $w = u \cdot g^h$, the equation $h = H_2(M||w, N)$ is verified. If valid, accepts the signature; otherwise rejected.

2.4. Secret Sharing

Secret sharing is a fundamental cryptographic primitive allowing a dealer to distribute a secret s among n participants such that only a subset of at least t shares (the threshold) can reconstruct the secret. Based on Lagrange polynomial interpolation, Shamir [30] proposed Shamir's Secret Sharing (SSS), which remains a cornerstone of modern cryptographic protocols. SSS comprises two primary algorithms: (1) $share(s, t, x_i) \rightarrow y_i$, which generates n shares where x_i is the public identifier and y_i is the secret share for the i -th participant; and (2) $reconstruct(x_i, y_{i=1}^k) \rightarrow s$, which recovers the secret s given any k valid shares, provided that $k \geq t$.

- **Secret Sharing Algorithm:** The secret owner randomly selects $t - 1$ values: $a_1, \dots, a_{t-1} \in \mathbb{Z}_p$, where p is a large prime number greater than secret, and then constructs a secret sharing polynomial: $f(x) = s + a_1x + \dots + a_{t-1}x^{t-1} \bmod p$. For each given public value x_i , the secret owner computes the corresponding secret share $y_i = f(x_i)$ and sends y_i to the respective entity.
- **Secret Reconstruction Algorithm:** After combining t secret shares $y_1, \dots, y_t$, the secret-sharing polynomial $f(x)$ can be reconstructed using Lagrange interpolation, with the detailed reconstruction process given in (1)

$$f(x) = \sum_{i=1}^t y_i \prod_{j=1, j \neq i}^t \frac{x - x_j}{x_i - x_j} \bmod p. \quad (1)$$

Since $s = f(0)$, the secret s can be directly reconstructed using (2) as follows:

$$s = \sum_{i=1}^t y_i \prod_{j=1, j \neq i}^t \frac{-x_j}{x_i - x_j} \bmod p. \quad (2)$$

3. Design Method

Blockchain addresses must ensure pseudonymity to protect the privacy of transacting parties. To this end, Dong et al. [20] proposed an SM9-based anonymous transaction scheme employing linkable ring signatures and a key derivation mechanism. However, in their scheme, derived addresses are cryptographically independent of the source address and are generated autonomously by users, rendering the system vulnerable to Sybil attacks. To address this limitation, this paper proposes a novel blockchain address derivation mechanism based on Shamir's secret sharing. While users retain the autonomy to generate derived addresses from their master private keys, these addresses exhibit homomorphic verifiability. This property enables the detection of a shared origin in scenarios such as e-voting, thereby effectively mitigating Sybil attacks. Crucially, to preserve the required pseudonymity of blockchain addresses, the verification process solely proves whether multiple addresses are derived from the same source, without disclosing the actual identity of the source address.

3.1. Algorithm Definition and System Model

In view of the characteristics of account addresses in blockchains, this section proposes a new blockchain address generation scheme based on the SM9 and Shamir's secret sharing scheme, including a derived address generation scheme. This scheme Π consists of four polynomial algorithms: $\Pi = (\text{Setup}, \text{AddrGen}, \text{DeriAddrGen}, \text{Trace})$, and its formal definition is as follows:

- **Setup**(1^λ) $\rightarrow$ (**mpk**, **msk**, **PP**): This algorithm is probabilistic. Taking a security parameter λ as input, the KGC(Key Generation Center) executes the system setup algorithm to output the system master public key (mpk), master secret key (msk), and public parameters (PP). These public parameters, which include a cryptographic hash function H , serve as input to the subsequent algorithms. Notably, the master secret key (msk) is securely maintained by the KGC.
- **AddrGen**(**msk**, **ID**) $\rightarrow$ (**sk_{ID}**, **sa**): This is a probabilistic algorithm. KGC takes as input the master secret key (msk) and a user's initial public key(identifier ID), then outputs the user's private key sk_{ID} . For guaranteeing blockchain address unpredictability and mitigating insider threats, user adopt self-selected cryptographic hash function H_3 . The user's primary address is computed as $sa = H_3(ID)$, where sa serves as the source address for all derived addresses.
- **DeriAddrGen**(**sa**, **n**, **DPK**) $\rightarrow$ (**DA**): This is a probabilistic algorithm. The user inputs user's account address, total number of derived accounts n and set of derived account public keys $DPK = (dpk_1, dpk_2, \dots, dpk_n)$. Outputs derived addresses $DA = (da_1, da_2, \dots, da_n)$. Let ssa_{dai} denotes the original address's secret sharing, dai denotes the index of a derived account in DA, that is share index, then $da_{dai} = H_3(dpk_i) || ssa_{dai} || dai$. Since ssa_{dai} does not reveal the user's real identity, da_{dai} satisfies pseudonymity.
- **Trace**(**da₁**, **da₂**, **da₃**) $\rightarrow$ **0/1**: This algorithm is deterministic. For input addresses da_1, da_2, da_3 , it outputs a bit $b \in \{0, 1\}$, where $b = 1$ indicates that addresses share a common origin, and $b = 0$ otherwise.

The detailed procedure of the proposed scheme is shown in Figure 1:

1. System Initialization: The KGC runs Setup to generate the system parameters.
2. Key Generation: The KGC runs the AddrGen algorithm, using the user's identity string (ID) as the public key, to compute and issue the corresponding user private key and source address.
3. Multi-User Derivation: Multiple users (User1, User2, ...) can independently execute the DeriAddrGen algorithm. Based on their respective source addresses, this algorithm generates hierarchical derived public-private key pairs and derived addresses [20,31].
4. Tracing and Clustering: When a dispute arises (e.g., vote manipulation), the Verifier collects all addresses to be verified and executes the Trace algorithm. Based on the common-origin determination results, this algorithm automatically clusters the intermingled addresses into groups, thereby accurately identifying which addresses belong to different aliases of the same malicious user and effectively thwarting Sybil attacks.

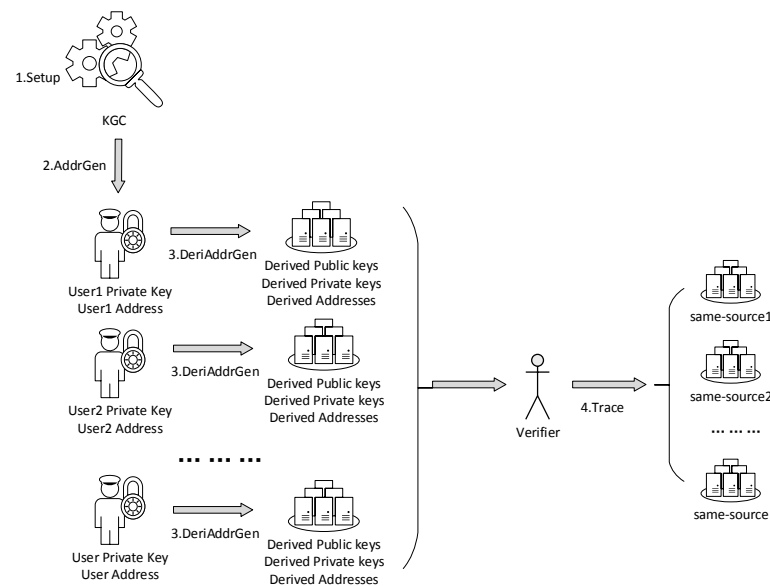


Figure 1. Procedure for homomorphically verifiable blockchain address generation.

3.2. Security Model

This section formalizes a security model for our scheme, incorporating critical security and privacy guarantees: unpredictability and traceability.

Definition 1. Unpredictability: We say the scheme satisfies unpredictability if no efficient adversary $\mathcal{A}$ can win the unpredictability game with non-negligible advantage $ADV_{\mathcal{A}}^{NPR}$.

- (1) **System Setup:** challenger $\mathcal{S}$ takes the security parameter λ and a polynomial $f(\cdot)$ as input, constructs a maximal original public key set $U_{\lambda} = (ID_1, \dots, ID_{f(\lambda)})$, and derives the system parameters (mpk, msk, PP) . The public components (mpk, PP) are revealed to $\mathcal{A}$.
- (2) **Suery Phase:** $\mathcal{A}$ may submit adaptive queries to the following oracles:
 - (i) **Source Address Query Oracle** $\mathcal{O}_{Addr}(ID) \rightarrow \mathbf{addrID}$: Given a user's original public key ID as input from $\mathcal{A}$, if $ID \in U_{\lambda}$, the challenger executes the address generation algorithm $AddGen$ to compute the corresponding address $addr_{ID}$ and returns it to $\mathcal{A}$. Otherwise, it returns $\perp$. Let L_{addr} denote the query set of $\mathcal{O}_{Addr}$, where each entry is a tuple $(ID, addrID)$.
 - (ii) **Derived Address Query Oracle** $\mathcal{O}_{DAddr}(ID, index) \rightarrow \mathbf{daddr}$: Given a user's original public key ID and a derivation index $index$ as input from $\mathcal{A}$, if $ID \in U_{\lambda}$, the challenger computes the $index$ -th derived address $daddr$ associated with ID and returns it to $\mathcal{A}$. Otherwise, it returns $\perp$. Let L_{daddr} denote the query set of $\mathcal{O}_{DAddr}$, where each entry is a triple $(ID, index, daddr)$.
- (3) **Challenge Phase:** The challenger selects an unqueried ID^* , arbitrarily chooses a derivation index $index^*$, generates the genuine source address $addr_0$ and the derived address $daddr_0$ corresponding to $index^*$, then randomly generates $addr_1$ and $daddr_1$. It flips a coin $b \in \{0, 1\}$ and sends $addr_b$ and $daddr_b$ to $\mathcal{A}$.
- (4) **Guess Phase:** $\mathcal{A}$ outputs $b' \in \{0, 1\}$.

The advantage of $\mathcal{A}$ in the above game is defined as $ADV_{\mathcal{A}}^{NPR} = |\Pr(b = b') - \frac{1}{2}|$.

Definition 2. Traceability: We say the scheme satisfies traceability if no efficient adversary $\mathcal{A}$ can win the traceability game with non-negligible advantage $ADV_{\mathcal{A}}^{TRAC}$.

- (1) **System Setup:** The same as the system setup phase for unpredictability.
- (2) **Suery Phase:** The same as the suery phase for unpredictability.

- (3) **Challenge Phase:** The adversary $\mathcal{A}$ submits a set of derived addresses (da_1, da_2, da_3) . If all derived addresses originate from the same user source address, then $b = 1$; otherwise, $b = 0$. If the challenger $\mathcal{S}$ satisfies $\text{Trace}(da_1, da_2, da_3) = b$, then $\mathcal{S}$ is said to have won the traceability game.

The advantage of $\mathcal{A}$ in the above game is defined as $\text{ADV}_{\mathcal{A}}^{\text{TRAC}}$.

Definition 3. Sybil Resistance: A scheme is said to be Sybil-resistant if, for any polynomial-time adversary $\mathcal{A}$, $\mathcal{A}$'s advantage in winning the Sybil resistance game defined below is negligible:

- (1) The adversary $\mathcal{A}$ derives m addresses $T = \text{addr}_1, \text{addr}_2, \dots, \text{addr}_m$ from a registered source address Addr_{src} , where each addr_i satisfies $\text{addr}_i \leftarrow \text{DeriveAddrGen}(\text{sk}_{\text{src}}, i)$
- (2) The adversary is said to win the Sybil resistance game if, during the identity verification process, the system recognizes at least three distinct addresses $\text{addr}_i, \text{addr}_j, \text{addr}_t (i \neq j \neq t)$ in T as mutually independent legitimate identities.

3.3. Scheme Construction

Building upon the algorithmic framework defined in Section 3.1, this section presents the concrete construction of the proposed scheme. To accommodate the decentralized nature of blockchain networks, we employ the SM9 algorithm for asymmetric key generation. SM9 is particularly well-suited for blockchain environments due to its certificate-free architecture, simplified key management, and low operational overhead [32]. Additionally, the generation of derived keys leverages the paradigm of hierarchical cryptography [31]. The core innovation of our scheme lies in deriving addresses based on Shamir's secret sharing, rather than merely computing the hash of the derived public keys. This approach enables the source traceability of derived addresses, thereby effectively mitigating Sybil attacks. Crucially, to preserve the strict pseudonymity of blockchain addresses, the derivation process incorporates the hash of the source address. This design ensures that the verification protocol can solely prove whether multiple addresses originate from a common source, without disclosing the actual identity of the source account.

- **System Setup ($\text{mpk}, \text{msk}, \text{PP}$)** $\leftarrow \text{Setup}(1^\lambda)$: Given the security parameter λ , the KGC first follows the SM9 digital signature algorithm to select an asymmetric bilinear group $\mathcal{BP} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, N)$ based on λ , where N is a large prime number and $N > \lambda$. Generators $P_1 \in \mathbb{G}_1, P_2 \in \mathbb{G}_2$ are chosen. A random number $\alpha \in \mathbb{Z}_N^*$ is selected, along with two cryptographic hash functions H_1, H_2 , which map $\{0, 1\}^*$ and $\mathbb{Z}_N^*$ to $\mathbb{Z}_N^*$. The following values are computed and published: $P_{\text{pub}} = \alpha P_2, g = e(P_1, P_{\text{pub}})$ in $\mathbb{G}_T$, a one-byte private key generation function identifier hid . The system's master public key (mpk) and master private key (msk) are then: $\text{mpk} = P_{\text{pub}}, \text{msk} = \alpha$. The system parameters are $\text{PP} = (\mathcal{BP}, P_1, P_2, H_1, H_2, \text{hid})$.
- **User Key Pair and Address Generation ($\text{sk}_{\text{ID}}, \text{sa}$)** $\leftarrow \text{AddrGen}(\text{msk}, \text{ID})$: Given a user identifier $\text{ID} \in \{0, 1\}^*$, to generate the user's key pair, the KGC computes $t_1 = H_1(\text{ID}) + \alpha$ in the finite field $\alpha \in \mathbb{Z}_N^*$. If $t_1 = 0$, the system master private key α must be regenerated, the system master public key recomputed and published, and all existing user private keys updated. Otherwise, the KGC computes $t_2 = \alpha \cdot t_1^{-1}$, $\text{sk}_{\text{ID}} = t_2 \cdot P_1, \text{sa} = H_3(\text{ID})$, where H_3 is a cryptographic hash function selected by the user, which map $\{0, 1\}^*$ and $\mathbb{Z}_N^*$ to $\mathbb{Z}_N^*$. Outputting sk_{ID} as the private key corresponding to the public identifier ID and sa as the user's account address which also serves as the source address for derived addresses.
- **Derived Address Generation (DA)** $\leftarrow \text{DeriAddrGen}(\text{sa}, n, \text{DPK})$: Select a random number $a \in \mathbb{Z}_p$ where p is a large prime number greater than $H_3(\text{sa})$, and construct the polynomial $f(x) = H_3(\text{sa}) + ax \bmod p$. For the set of n derived

addresses $DA = \{da_1, da_2, \dots, da_n\}$, take $ssa_i = f(i) = H_3(sa) + ai \bmod p$, then $da_i = da_i = H_3(dpk_i) || ssa_i || i$.

- **Same-origin Trace 0/1** $\leftarrow \text{Trace}(da_1, da_2, da_3)$: Given any addresses da_1, da_2, da_3 , input them into the address same-origin verification algorithm. If the verification passes, output 1; otherwise, output 0. The address same-origin verification algorithm proceeds as follows:
 - (1) Extract secret sharing ssa_i and share index i from each da_i ;
 - (2) Arbitrarily select two pairs of derived addresses $(da_a, da_b), (da_a, da_c)$, and compute:

$$v_1 = ssa_a \frac{-b}{a-b} + ssa_b \frac{-a}{b-a} \quad v_2 = ssa_a \frac{-c}{a-c} + ssa_c \frac{-a}{c-a} \quad (3)$$

- (3) Compare whether v_1 and v_2 are equal, If they are equal, return 1; otherwise, return 0.

3.4. Security Analysis of This Program

This section formally proves that the proposed scheme satisfies the essential security properties of **uniqueness, unpredictability, pseudonymity, traceability** and **Sybil-resistant**.

Uniqueness. We first demonstrate the uniqueness of the generated addresses. In the proposed scheme, the source account address is computed as $sa = H(ID)$. Given that the user's ID serves as a globally unique identifier and the hash function H exhibits strong collision resistance [33], the source address sa is inherently unique. A derived account address comprises three components: a derived public key, a secret share of the source address, and a share index. The uniqueness of the derived address fundamentally relies on the derived public key. In elliptic curve-based blockchain systems, the key space is defined by a cyclic group of large prime order. The probability of two distinct users independently generating the identical private key (and consequently the same public key) is computationally negligible. Furthermore, even when deterministic derivation algorithms are utilized, the integration of unique derivation parameters ensures that each derived public key remains distinct. Consequently, the derived public key is globally unique with overwhelming probability, thereby guaranteeing the strict uniqueness of the derived account address.

Pseudonymity. Pseudonymity ensures that a user's real-world identity cannot be directly inferred from their blockchain address, although it remains traceable given specific auxiliary information. In the proposed scheme, the source address is constructed as $H_3(ID)$, where ID represents the user's unique identity identifier. Similarly, the derived address incorporates $H_3(dpk)$, where dpk denotes the unique public key of the derived account. Due to the pre-image resistance of the cryptographic hash function, it is computationally infeasible to reverse-engineer the user's identity from either address, thereby preserving strict privacy against the general public. However, for authorized verifiers possessing the auxiliary information (i.e., the user's ID and the derived public key dpk), the linkage between the address and the real identity can be deterministically verified. Consequently, the proposed scheme rigorously satisfies the cryptographic definition of pseudonymity.

Theorem 1. *If the hash function $H_3 : \{0, 1\}^* \rightarrow \{0, 1\}^n$ is a random oracle, then the scheme proposed in this paper satisfies unpredictability.*

Proof. $\mathcal{S}$ establishes the following lists and initializes them as empty: list L_H stores hash queries of $H_3(x)$ (where elements in the list are tuples (x, h)), list L_{addr} stores source address queries, and list L_{daddr} stores derived address queries.

- (1) **System Setup:** $\mathcal{S}$ implicitly sets the system master secret key $msk = \alpha$. Subsequently, $\mathcal{S}$ generates the corresponding initial system data as follows: Randomly selects generators $P_1 \in \mathbb{G}_1$ and $P_2 \in \mathbb{G}_2$, computes $P_{pub} = \alpha P_2$, and $g = e(P_1, P_{pub})$. H_1 , H_2 , and H_3 are treated as random oracles controlled by $\mathcal{S}$, where H_1 and H_2 are public parameters that can be sent to the adversary $\mathcal{A}$, while H_3 remains undisclosed.
- (2) **Suery Phase:** The attacker $\mathcal{A}$ can adaptively access the random oracle multiple times and perform hash queries, source address queries, and derived address queries.
 - (i) **Hash Query Oracle:** $\mathcal{A}$ can query $\mathcal{S}$ with either a user ID, a user source address sa , or a derived public key dpk_i . $\mathcal{S}$ maintains a list L_H to record all queried values. If the hash value corresponding to the input exists in L_H , $\mathcal{S}$ returns it; otherwise, $\mathcal{S}$ generates a random number, returns it to $\mathcal{A}$, and stores the corresponding tuple in L_H .
 - (ii) **Source Address Query Oracle:** $\mathcal{A}$ may query $\mathcal{S}$ with any original user $ID \in U_\lambda$ for address generation. If $ID \notin L_{addr}$, $\mathcal{S}$ first checks list L_H —if a corresponding tuple exists, its value is returned to $\mathcal{A}$; otherwise, $\mathcal{S}$ computes $sa = H_3(ID)$, returns sa as the user's source address to $\mathcal{A}$, and stores it in both L_H and L_{addr} .
 - (iii) **Derived Address Query Oracle:** If $ID, index \notin L_{daddr}$, select a random number $a \in \mathbb{Z}_p$, compute $sa = H_3(ID)$, construct the polynomial $f(x) = H_3(sa) + ax$, then derive $da = H_3(dpk) \parallel f(index) \parallel index$. Return da as the derived address to the adversary $\mathcal{A}$, and store it in L_{daddr} .
- (3) **Challenge Phase:** The challenger selects $(ID^*, index^*)$, where ID^* is an unqueried user ID and $index^*$ is an arbitrary derivation index. It computes: $addr_0 = H_3(ID^*)$, $daddr_0 = H_3(dpk) \parallel H_3(sa) + a \cdot index \parallel index$, then randomly generates $addr_1$ and $daddr_1$. A bit $b \in \{0, 1\}$ is uniformly chosen, and $(ID^*, index^*, addr_b, daddr_b)$ is sent to $\mathcal{A}$.
- (4) **Guess Phase:** The attacker $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$. If $b = b'$, then $\mathcal{A}$ wins the game. Let $q(\lambda)$ be the number of queries made by $\mathcal{A}$ (where λ is the security parameter). Define the event Bad as “ $\mathcal{A}$ queries ID^* via a hash query before the challenge phase” (where ID is chosen at random). Since ID^* is a value that has never been queried at the time of the challenge, in effect, once the adversary's query set L_H is fixed, ID^* is chosen uniformly at random from $\mathcal{X} \setminus L_H$ (where $\mathcal{X}$ is the domain of ID^* , typically $|\mathcal{X}| = 2^\lambda$). Therefore,

$$\Pr(\text{Bad}) \leq \frac{|L_H|}{|\mathcal{X}| - |L_H|} \leq \frac{q}{|\mathcal{X}| - q} \leq \frac{2q}{|\mathcal{X}|} \quad (4)$$

For a sufficiently large λ (where $q \ll |\mathcal{X}|$), since $|\mathcal{X}|$ is exponential, $\Pr(\text{Bad})$ is negligible.

If $\neg \text{Bad}$, then $\mathcal{A}$ has never queried ID^* . In this case, $H_3(ID^*)$ is uniformly random in $\mathcal{A}$'s view and independent of $addr_1$. Thus, $\mathcal{A}$ cannot obtain any information about b , which implies $\Pr(b' = b | \neg \text{Bad}) = \frac{1}{2}$. Then,

$$\Pr(b' = b) = \Pr(b' = b | \text{Bad})\Pr(\text{Bad}) + \frac{1}{2}(1 - \Pr(\text{Bad})) \leq \frac{1}{2} + \frac{1}{2}\Pr(\text{Bad}). \quad (5)$$

Similarly, we can obtain the lower bound $\Pr(b' = b) \geq \frac{1}{2} - \frac{1}{2}\Pr(\text{Bad})$. Therefore,

$$|\Pr(b' = b) - \frac{1}{2}| \leq \frac{1}{2}\Pr(\text{Bad}) \leq \text{negl}(\lambda). \quad (6)$$

The advantage of $\mathcal{A}$ in winning the unpredictability game is negligible. $\square$

Theorem 2. *If Lagrange interpolation is correct and the hash function H_3 is collision-resistant, then the scheme proposed in this paper satisfies traceability.*

Proof. If there exists a polynomial-time adversary $\mathcal{A}$ such that $ADV_{\mathcal{A}}^{Trace} = \epsilon$ is non-negligible, then there exists a polynomial-time algorithm capable of finding a collision for the hash function H_3 with probability at least ϵ (i.e., two distinct inputs $x \neq y$ such that $H_3(x) = H_3(y)$).

Similar to the unpredictability proof, the challenger $\mathcal{S}$ and the adversary $\mathcal{A}$ execute the system setup phase along with hash query, source address query, and derived address query phases, where $\mathcal{A}$ can obtain the source address corresponding to any user ID and any number of derived addresses through queries.

The construction of the derived address for a given ID is $da = H_3(dpk) || f(index) || index$ where dpk is the derived account public key, $index$ is the derived address index, $f(x) = H_3(sa) + ax \bmod p$, sa is the source address corresponding to the derived address, and $a \in \mathbb{Z}_p$ is a fixed random value, where p is a prime greater than $H_3(sa)$. Considering only the second and third parts of the derived address construction, $f(index)$ and $index$, a set of derived addresses da_1, da_2, da_3 submitted by $\mathcal{A}$ can be written as $f(i_1) || i_1, f(i_2) || i_2, f(i_3) || i_3$. From the construction, we have

$$f(i_1) = H_3(sa_{i_1}) + ai_1 \quad f(i_2) = H_3(sa_{i_2}) + ai_2 \quad f(i_3) = H_3(sa_{i_3}) + ai_3 \quad (7)$$

Substituting these into the Lagrange interpolation formula yields

$$\begin{aligned} v_1 &= \frac{H_3(sa_{i_1}) - H_3(sa_{i_2})}{i_1 - i_2} + 1 + a \\ v_2 &= \frac{H_3(sa_{i_1}) - H_3(sa_{i_3})}{i_1 - i_3} + 1 + a \end{aligned} \quad (8)$$

The winning condition $v_1 = v_2$ implies that

$$\frac{H_3(sa_{i_1}) - H_3(sa_{i_2})}{i_1 - i_2} = \frac{H_3(sa_{i_1}) - H_3(sa_{i_3})}{i_1 - i_3} \quad (9)$$

Since i_1, i_2, i_3 are pairwise distinct, the above equation holds in two cases:

- Case A: $H_3(sa_{i_1}), H_3(sa_{i_2}), H_3(sa_{i_3})$ are not all equal, but the above equation holds by coincidence. The probability of this case is negligible because, under the random oracle model, the values $H_3(sa_i)$ are independently and uniformly distributed.
- Case B: $H_3(sa_{i_1}) = H_3(sa_{i_2}) = H_3(sa_{i_3})$

If $\mathcal{A}$ wins with a non-negligible probability ϵ , then the probability of Case B must also be non-negligible. Under the distinct-source condition, namely

$$sa_{i_1} \neq sa_{i_2} \neq sa_{i_3} \quad (10)$$

if ϵ is non-negligible, this would break the collision resistance of H_3 with non-negligible probability. Therefore, $ADV_{\mathcal{A}}^{Trace} = \epsilon$ is negligible.

When da_1, da_2, da_3 share the same source, the recovered secret is the hash value of sa , thus, the verification process does not expose the user's source address, satisfying the unpredictability of blockchain addresses. Same-source detection only proves whether the derived addresses originate from the same source and does not reveal the specific source address. $\square$

Theorem 3. *If the scheme satisfies traceability, then it is Sybil-resistant.*

Proof. Suppose the scheme is not Sybil-resistant. Then there exists an adversary $\mathcal{A}$ that causes the system to misidentify three addresses $addr_i, addr_j, addr_t$ derived from the same source address $Addr_{src}$ as independent identities. By Definition 2, this implies that the tracing algorithm Trace fails to link all three to the same source address. However, this

directly contradicts traceability—because traceability requires Trace to correctly attribute all addresses derived from the same source address to that single source. Contradiction. $\square$

4. Performance Analysis

4.1. Theoretical Analysis

Addresses in traditional blockchains (e.g., Bitcoin and Ethereum) are based on Elliptic Curve Cryptography (ECC) and the Public Key Infrastructure (PKI). To address the inherent efficiency bottlenecks and management complexities of PKI in large-scale applications, ref. [19] successfully proposed a method for generating blockchain addresses using an Identity-Based Cryptography (IBC) scheme (the SM9 algorithm); however, its primary focus is limited to basic feasibility and efficiency. Drawing on the concept of hierarchical cryptography, ref. [20] designed a blockchain anonymous transaction scheme based on the SM9 algorithm, which protects recipient address privacy but fails to consider the possibility of multiple derived addresses being used collectively to launch Sybil attacks. This paper primarily focuses on how users with legitimate identities can generate derived addresses within SM9-based blockchain address generation schemes, thereby preventing participants from exploiting derived addresses to launch Sybil attacks. Table 1 presents a comparison of the four schemes:

Table 1. Comparison of different schemes.

Compared Scheme	Ours	Ref. [19]	Ref. [20]	ECDSA (Bitcoin)
Identity-based	✓	✓	✓	×
Sender Address Privacy	✓	✓	✓	✓
Recipient Address Privacy	✓	×	✓	×
Sybil-resistant	✓	×	×	×

Table 2 presents a theoretical comparison of the computing costs among the four schemes, and Table 3 presents a theoretical comparison of their communication costs.

Table 2. Comparison of computing costs. n is the number of driven members, T_G is the time for scalar multiplication in G , T_{inv} is the time for modular inverse in $\mathbb{Z}_N^*$, T_h is the average time to compute the Hash function, T_f is the time for polynomial operations over finite fields. Other computational overheads are relatively small compared to the above operations and are therefore omitted from the statistics.

Compared Scheme	Ours	Ref. [19]	Ref. [20]	ECDSA (Bitcoin)
System Setup	$T_G + T_{inv} + T_h$	$T_G + T_{inv} + T_h$	$T_G + T_{inv} + T_h$	×
User Key Pair and Address Generation	$T_G + T_{inv} + T_h$	$T_G + T_{inv} + T_h$	$T_G + T_{inv} + T_h$	$T_G + T_h$
Derived Address Generation	$T_f + nT_h$	×	nT_h	×
Same-origin Trace	nT_h	×	×	×

Bitcoin employs an elliptic curve-based key generation mechanism for address derivation, and both the proposed scheme and the approaches in [19,20] utilize identity-based cryptography. Among these identity-based solutions, only our scheme and the one proposed by Dong et al. [20] support the generation of derived user addresses. As illustrated in Table 2, identity-based address generation inherently incurs a relatively higher computational overhead compared to traditional ECC-based methods. During the system setup and user key extraction phases, the computational cost of our scheme is equivalent to that of [19,20]. To enable address traceability, our scheme incurs a marginal increase in compu-

tational overhead during the derived address generation phase compared to [20]. However, the proposed scheme achieves 100% source tracing accuracy for derived addresses, effectively preventing participants from exploiting these derived addresses to launch Sybil attacks. Regarding storage and communication costs, as detailed in Table 3, the key and address lengths in our scheme remain virtually identical to those in [20]. Ultimately, by substituting the standard derived user index with a secret share index, our scheme successfully achieves effective address traceability without significantly compromising overall efficiency.

Table 3. Comparison of communication costs. n is the number of driven members, $|\mathbb{G}|$ is the bit lengths of elements in $\mathbb{G}$, $|Secret_i|$ is the bit lengths of i -th secret share.

Compared Scheme	Ours	Ref. [19]	Ref. [20]	ECDSA (Bitcoin)
System pk size	$ \mathbb{G} $	$ \mathbb{G} $	$ \mathbb{G} $	$\times$
User pk size	$ ID_{user} + Secret_i $	$ ID $	$ ID_{user} + ID_{deriveduser} $	$ \mathbb{G} $
User sk size	$ \mathbb{G} $	$ \mathbb{G} $	$ \mathbb{G} $	N
User address size	$ ID_{user} + Secret_i $	$ ID $	$ ID_{user} + ID_{deriveduser} $	$ \mathbb{G} $

4.2. Experimental Analysis

This subsection evaluates the proposed scheme through experimental simulations. The algorithms were implemented in Go using the Type-A pairing parameters from the PBC (Pairing-Based Cryptography) library (version 0.5.14). The standard security parameters adopted are a 160-bit group order r and a 512-bit base field q , providing approximately 80 bits of security. The Shamir secret sharing module implements polynomial interpolation and reconstruction over a 256-bit prime field. All share generation and secret reconstruction operations are performed over this finite field, providing approximately 128 bits of security strength. All performance and security experiments are conducted under this security parameter configuration. The experiments were conducted on a local machine equipped with an Intel Core i5-4570 processor, 8 GB of RAM, and a 64-bit Windows 7 operating system.

Let M denote the number of addresses derived from a single source, and N the total number of derived addresses. The formula for the proportion of same-origin addresses P is given by:

$$P = M/N \quad (11)$$

Table 4 presents a comparison of the accuracy and time overhead of derived address tracing for varying numbers of blockchain addresses. As shown in the table, the proposed scheme achieves 100% accuracy in derived address tracing. The tracing time is correlated with the number of addresses involved in the tracing process, but is independent of the proportion of same-source addresses. Figure 2a illustrates the increasing trend in the time overhead of tracing operations as the total number of addresses involved in tracing grows, under a fixed same-source address ratio (50%). The time overhead increases with the total number of addresses involved in the tracing process. Figure 2b illustrates the variation in tracing time overhead as the same-source address ratio increases, with a fixed total number of addresses involved in tracing (1000). Changes in the same-source address ratio have no impact on the time overhead of tracing operations.

Figure 3 illustrates the address (200) generation time across various schemes as the number of users scales, comparing our proposed scheme with the approaches in [19,20] and the standard Bitcoin address generation mechanism (based on ECDSA). The empirical results align closely with our theoretical analysis. Specifically, the total time required for ad-

dress generation scales proportionally with the number of users. The ECDSA-based scheme incurs the lowest computational overhead, primarily because it avoids complex operations such as modular inversion during the address derivation process. In contrast, our proposed scheme exhibits the highest time overhead. This is attributed to the additional polynomial operations over finite fields required to generate the derived addresses. Nevertheless, this performance trade-off is fundamentally justified, as it uniquely enables derived address traceability and effectively mitigates Sybil attacks—critical security features that are absent in the baseline schemes.

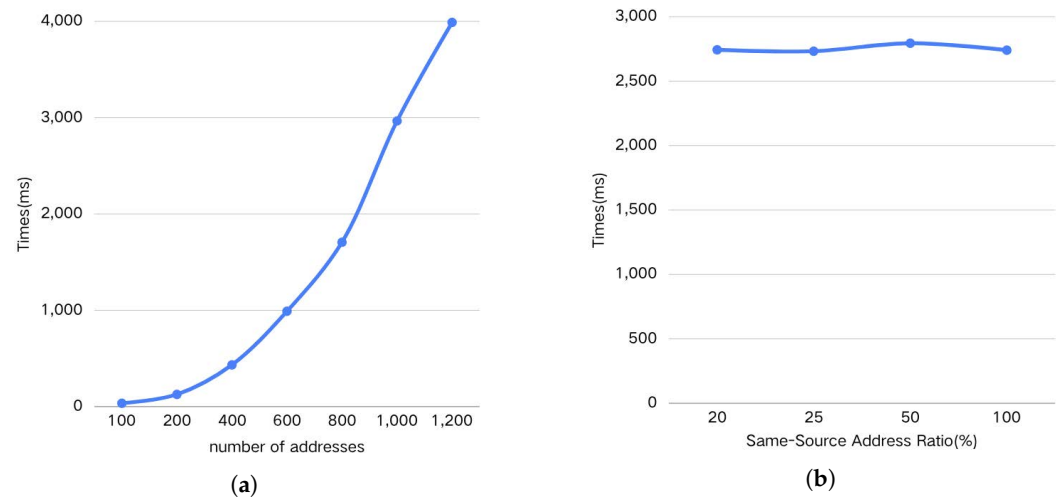


Figure 2. Performance evaluation of tracing operations. (a) Time overhead vs. total number of traced addresses under a fixed same-source address ratio (50%). (b) Time overhead vs. same-source address ratio under a fixed number of traced addresses (1000).

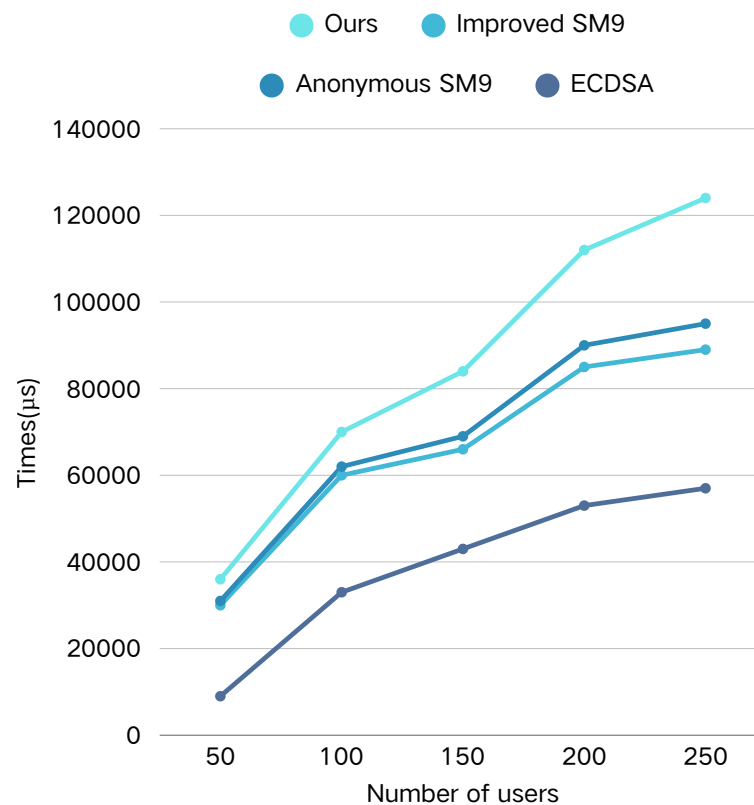


Figure 3. Comparison of time consumption for blockchain addresses (200) generation algorithms under different scheme standards.

Table 4. Accuracy and time overhead of derived address tracing.

Number of Source Addresses	Number of Derived Addresses per Source Address	Proportion of Same-Source Addresses (%)	Accuracy (%)	Time Overhead (μ s)
1	100	100	100	31,001.70
3	100	33.3	100	244,013.90
5	100	20	100	665,038.00
1	500	100	100	698,040.00

5. Conclusions

In blockchain networks, derived user addresses are generally cryptographically independent of their source addresses, thereby creating opportunities for Sybil attacks. To address this issue, this paper proposes a novel blockchain address derivation scheme based on the SM9 identity-based cryptographic standard. This scheme achieves origin traceability through provable linkability. We formalize the system and security models and present the concrete construction of the proposed scheme. Although both our scheme and the method in [19] employ SM9 as the core cryptographic primitive, our scheme uniquely integrates the address derivation functionality to support anonymous blockchain transactions, thereby expanding its applicability across diverse scenarios. Compared with the ring signature-based address derivation scheme in [20], our scheme generates addresses of comparable length, but its key advantage lies in supporting origin traceability with 100% tracing accuracy. This feature effectively prevents attackers from launching Sybil attacks via derived addresses in applications such as e-voting. Although the introduction of the traceability mechanism results in slightly higher computational overhead compared to the schemes in [19,20], this performance trade-off is essential for providing robust security guarantees. Optimizing the computational efficiency of the derivation process remains a promising direction for future research.

Author Contributions: Conceptualization, J.T.; methodology, J.T.; software, J.T.; validation, F.S.; resources, C.X.; writing—original draft preparation, J.T.; writing—review and editing, C.X.; funding acquisition, C.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant Nos.62472227 and 61902189.

Data Availability Statement: The code data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ECDSA	Elliptic Curve Digital Signature Algorithm
PKI	Public Key Infrastructure
IBC	Identity-Based Cryptography
KGC	Key Generation Center

References

1. Nakamoto S. Bitcoin: A Peer to Peer Electronic Cash System. 2008. Available online: <https://bitcoin.org/bitcoin.pdf> (accessed on 2 July 2026).
2. Bodkhe, U.; Tanwar, S.; Parekh, K.; Khanpara, P.; Alazab, M. Blockchain for industry 4.0: A comprehensive review. *IEEE Access* **2020**, *8*, 79764–79800. [CrossRef]

3. Blythe, J.M.; Johnson, S.D. A systematic review of crime facilitated by the consumer internet of things: Jm blythe, sd johnson. *Secur. J.* **2021**, *34*, 97–125.
4. Kumar, A.; Rathee, G.; Kerrache, C.A.; Bilal, M.; Gadekallu, T.R. A secure architectural model using blockchain and estimated trust mechanism in electronic consumers. *IEEE Trans. Consum. Electron.* **2023**, *69*, 996–1004. [\[CrossRef\]](#)
5. Rathee, G.; Kerrache, C.A.; Calafate, C.T.; Halimi, M.S.. Secureblock: An ml-blockchain consumer-centric sustainable solution for industry 5.0. *IEEE Trans. Consum. Electron.* **2023**, *70*, 1452–1462.
6. Shamir, A. Identity-based cryptosystems and signature schemes. In *Workshop on the Theory and Application of Cryptographic Techniques*; Springer Berlin Heidelberg: Berlin, Heidelberg, Germany, 1984.
7. Zhu, X.; Fan, T. Research on application of blockchain and identity-based cryptography. *IOP Conf. Ser. Earth Environ. Sci.* **2019**, *252*, 042095. [\[CrossRef\]](#)
8. Kaaniche, N.; Laurent, M.; Levallois-Barth, C. Id-based user-centric data usage auditing scheme for distributed environments. *Front. Blockchain* **2020**, *3*, 17. [\[CrossRef\]](#)
9. Momeni, P. Ibe-ibe: Intent-based execution through identity-based encryption and auctions. *Cryptol. ePrint Arch.* 2025, Paper 2025/241. Available online: <https://eprint.iacr.org/2025/241> (accessed on 2 July 2026). [\[CrossRef\]](#)
10. Ermilov, D.; Panov, M.; Yanovich, Y. Automatic bitcoin address clustering. In Proceedings of the 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Cancun, Mexico, 18–21 December 2017; pp. 461–466. [\[CrossRef\]](#)
11. Rivest, R.L.; Shamir, A.; Tauman, Y. How to leak a secret. In *Advances in Cryptology—ASIACRYPT 2001, Gold Coast, Australia, 9–13 December 2001*; Boyd, C., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2001; Volume 2248, pp. 552–565. [\[CrossRef\]](#) [\[PubMed\]](#)
12. van Saberhagen, N. Cryptonote v 2.0. White Paper, 2013. Available online: <https://www.semanticscholar.org/paper/CryptoNote-v-2.0-Saberhagen/5bafdd891c1459ddfd22d71412d5365de723fb23#cited-papers> (accessed on 2 July 2026).
13. Sun, S.-F.; Au, M.H.; Liu, J.K.; Yuen, T.H. Ringct 2.0: A compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero. In *European Symposium on Research in Computer Security*; Springer International Publishing: Cham, Switzerland, 2017.
14. Yuen, T.H.; Sun, Sh.; Liu, J.K.; Au, M.H.; Esgin, M.F.; Zhang, Q.; Gu, D. Ringct 3.0 for blockchain confidential transaction: Shorter size and stronger security. In *International Conference on Financial Cryptography and Data Security*; Springer International Publishing: Cham, Switzerland, 2020.
15. Miers, I.; Garman, C.; Green, M.; Rubin, A.D. Zerocoin: Anonymous distributed e-cash from bitcoin. In Proceedings of the 2013 IEEE Symposium on Security and Privacy, Berkeley, CA, USA, 19–22 May 2013.
16. Parno, B.; Howell, J.; Gentry, C.; Raykova, M. Pinocchio: Nearly practical verifiable computation. *Commun. ACM* **2016**, *59*, 103–112. [\[CrossRef\]](#)
17. Sasson, E.B.; Chiesa, A.; Garman, C.; Green, M.; Miers, I.; Tromer, E.; Virza, M. Zerocash: Decentralized anonymous payments from bitcoin. In Proceedings of the 2014 IEEE Symposium on Security and Privacy, Berkeley, CA, USA, 18–21 May 2014.
18. GM/T 0044-2016; Technical Cryptography Standardization: SM9 Identity-Based Cryptographic Algorithms. Industry Standard, State Cryptographic Authority: Beijing, China, 2016. Available online: <https://std.samr.gov.cn/hb/search/stdHBDetailed?id=8B1827F140DEBB19E05397BE0A0AB44A> (accessed on 2 July 2026).
19. Wang, R.; Fan, H.; Liu, K. A blockchain address generation method based on the improved sm9 algorithm. *Electronics* **2025**, *14*, 585. [\[CrossRef\]](#)
20. Dong, J.; Li, C.; Shen, Z.; Hu, D.; Shen, Q.; Fang, Y.; Wu, Z. An anonymous blockchain transaction scheme based on sm9. *Sci. Sin. Informationis* **2025**, *55*, 1428–1446. [\[CrossRef\]](#)
21. Yang, X.; Yue, C.; Zhang, W.; Liu, Y.; Ooi, B.C.; Chen, J. Secudb: An in-enclave privacy-preserving and tamper-resistant relational database. *Proc. VLDB Endow.* **2024**, *17*, 3906–3919. [\[CrossRef\]](#)
22. Yang, X.; Zhang, R.; Yue, C.; Liu, Y.; Ooi, B.C.; Gao, Q.; Zhang, Y.; Yang, H. Vedb: A software and hardware enabled trusted relational database. *Proc. ACM Manag. Data* **2023**, *1*, 1–27. [\[CrossRef\]](#)
23. Yang, X.; Zhang, Y.; Wang, S.; Yu, B.; Li, F.; Li, Y.; Yan, W. Ledgerdb: A centralized ledger database for universal audit and verification. *Proc. VLDB Endow.* **2020**, *13*, 3138–3151. [\[CrossRef\]](#)
24. Al-Humaimedy, A.S. 3rvav: A three-round voting and proof-of-stake consensus protocol with provable byzantine fault tolerance. *Comput. Mater. Contin.* **2025**, *85*, 5207. [\[CrossRef\]](#)
25. Niu, J.; Gai, F.; Han, R.; Zhang, R.; Zhang, Y.; Feng, C. Crystal: Enhancing blockchain mining transparency with quorum certificate. *IEEE Trans. Dependable Secur. Comput.* **2022**, *20*, 4154–4168. [\[CrossRef\]](#)
26. Sun, Y.; Hou, R.; Yu, H. Using multi-dimensional quorums for optimal resilience in multi-resource blockchains. *Form. Asp. Comput.* **2024**, *36*, 1–28. [\[CrossRef\]](#)
27. Alpturer, K.; Doumanidis, C.; Zohar, A. Aetherweave: Sybil-resistant robust peer discovery with stake. *arXiv*, **2026**, arXiv:2603.23793.
28. Song, R. Linkdid: A privacy-preserving, sybil-resistant and key-recoverable decentralized identity scheme. *arXiv*, **2023**, arXiv:2307.14679.

29. Zhu, J.; Bai, S.; Hu, X.; Zhou, H.; Yang, G. Lapathzk: Parallel threshold path zk-snark leveraging high-dimensional lattices. *Inf. Sci.* **2026**, *755*, 123761. [[CrossRef](#)]
30. Shamir, A. How to share a secret. *Commun. ACM* **1979**, *22*, 612–613. [[CrossRef](#)]
31. Lai, J.; Huang, X.; He, D.; Guo, F. An efficient hierarchical identity-based encryption based on uppercase sm9. *Sci. Sin. Informationis* **2023**, *53*, 918–930.
32. Chen, Z.; Liu, L.; Wang, S. Review of sm9 identity authentication schemes and their applications. *Comput. Eng. Appl.* **2025**, *61*, 18–31.
33. Cai, X.Q.; Deng, Y.; Zhang, L.; Shi, J.C.; Chen, Q.; Zheng, W.L.; Liu, Z.Q.; Long, Y.; Wang, K.; Li, C.; Guo, M.Y. The principle and core technology of blockchain. *Chin. J. Comput.* **2021**, *44*, 84–131.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.