DIMSEPP: A Decentralized Identity Management System with Enhanced Privacy Protection

Yu Zhang*†, Zongbin Wang‡
*Northwestern Polytechnical University
†Beijing PushtimeTechnology Co., Ltd.
‡ Beijing Infosec Technologies Co., Ltd.

***Abstract*—This paper proposes DIMSEPP, a decentralized identity management system that enhances privacy while preserving blockchain verifiability. The system cryptographically enforces data minimal disclosure principles by storing attribute commitments on-chain and validating them through zero-knowledge proofs, allowing users to demonstrate attribute validity without revealing sensitive values.**

The architecture maintains full compatibility with existing DID standards through standard document structures and verification methods. Security analysis demonstrates provable guarantees under standard cryptographic assumptions. Practical evaluation confirms the system's efficiency for resource-constrained environments, supporting deployment in applications where both privacy and verifiability are essential.

I. Introduction

Decentralized Identity (DID) systems[1], [2], [3], [4], [5] have emerged as a promising alternative to traditional identity management, addressing critical issues of centralized control and single points of failure[6], [7]. Built on blockchain technology[8], [9], [10], [11], [12], [13], DIDs enable users to own and control their digital identities without relying on intermediaries[14], [15]. The concept gained significant traction with the W3C DID specification[16], establishing standards for interoperable decentralized identity solutions.

While DID systems offer advantages in user control and resilience, their architectural design introduces novel privacy challenges that demand urgent attention[17], [18]. A fundamental vulnerability stems from the storage of identity attributes on publicly accessible blockchain nodes. Unlike traditional systems where identity data resides in protected databases, DID implementations often record identity credentials on-chain, making them permanently visible to all network participants [7], [19]. This design, while ensuring verifiability, creates unprecedented risks of mass identity exposure.

Another critical privacy limitation involves the current approaches to credential presentation. Conventional DID systems typically require users to disclose complete credentials when proving authorization to service providers, violating the principle of minimal disclosure [20], [21], [22], [23], [24]. For instance, when applying for a loan that requires minimum annual income verification, existing systems force users to reveal their exact salary rather than simply proving they meet the threshold. This all-or-nothing approach unnecessarily exposes sensitive personal data, creating privacy risks and potential discrimination vectors.

These deficiencies become particularly acute in financial services and healthcare applications, where both regulatory requirements and user expectations demand strict privacy guarantees. The transparency of blockchain-based DID systems conflicts with data minimization principles enshrined in privacy regulations like GDPR [20], [21], while the coarse-grained authorization mechanisms fail to meet real-world needs for selective disclosure.

A. Related Work

Prior DID systems face critical privacy limitations in attribute management. The approach in [16] stores raw attribute values directly on-chain, exposing them to all network participants. While [17] enhances privacy by storing hashed attribute references that conceal the linkage between attributes and their owners, it still requires complete attribute disclosure during service authentication, failing to achieve true minimal disclosure.

[18] attempts to address this by allowing users to store only attribute predicates (e.g., "age $\geq$ 18") on-chain rather than values. However, this solution introduces new inefficiencies - requiring separate predicate records for each usage context (e.g., voting vs. employment checks) - while still leaking partial information through the published predicates themselves. Moreover, the repeated on-chain storage of context-specific predicates creates unnecessary blockchain bloat.

[25] proposes a blockchain-based trust management system for DID, which utilizes Secure Multi-Party Computation (SMPC) to protect feedback privacy. However, its privacy guarantees for data on-chain fundamentally rely on permissioned blockchain access control, requiring pre-authorized nodes to view data. This design inherently limits deployment in public DID ecosystems where fully decentralized verification is required.

B. Contributions

To overcome these limitations, we propose a Decentralized Identity Management System with Enhanced Privacy Protection (DIMSEPP), which makes the following key contributions to decentralized identity management:

- **Privacy-Preserving Architecture**: We propose the first DID system that combines *off-chain attribute storage* with *on-chain Pedersen commitments* [26], [27] and *zero-knowledge range proofs* [28], [29], [30], [31], resolving

the fundamental conflict between blockchain transparency and user privacy. This ensures sensitive attributes remain confidential while allowing verifiable predicate checks, achieving GDPR-compliant data minimization [20].

- **Practical Efficiency**: With *linear-time operations* and *constant-size DID records* (including both DID documents and verifiable credentials), DIMSEPP ensures practical deployability on resource-constrained devices such as mobile and IoT nodes. *Extensive experiments* confirm the system's efficiency in real-world settings (see V-B).
- **Backward Compatibility**: Designed to seamlessly integrate with W3C DID standards [16], our system requires only minimal adjustments to existing DID resolvers or wallets, while maintaining:
  - Uses standard *DID document structures*
  - Maintains existing *verification method* formats
  - Supports gradual adoption via hybrid modes (on-chain/off-chain credentials)
- **Provable Security**: We formally define and prove completeness (ensuring legitimate operations succeed) and security (including consistency, attribute-based indistinguishability and predicate-based indistinguishability).

II. Problem Statement and System Model

A. Problem Statement

We address the challenge of designing a decentralized identity system that satisfies the following requirements:

- All participants except Identity Providers (IdPs) are *semi-honest*: They follow protocol specifications but attempt to infer additional user information through metadata analysis or correlation attacks. IdPs are fully trusted for credential issuance but not for ongoing identity management.
- Identity documents (DID records and credential status) are stored on *publicly accessible* blockchain nodes to ensure verifiability and non-repudiation, while preventing single points of failure.
- User *privacy attributes* (e.g., age, salary, or affiliations) must remain confidential. Service Providers (SPs) can only verify predicate satisfaction (e.g., "Age $\geq$ 18") without learning specific values.

B. System Model

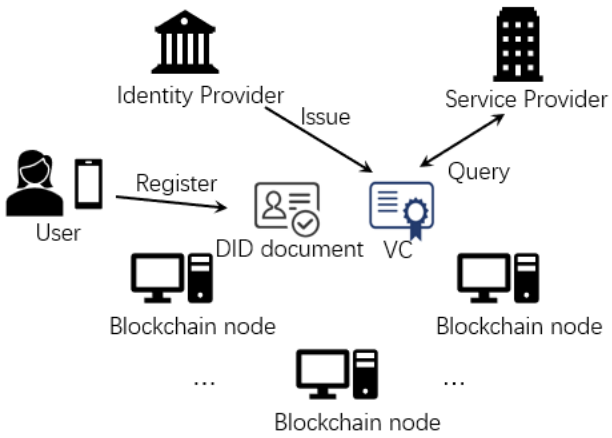

Fig. 1. System architecture

The DIMSEPP architecture (Fig. 1) extends standard DID protocols with cryptographic privacy enhancements. Users register DID documents containing public keys via blockchain transactions, while IdPs issue on-chain anchored, signed verifiable credentials (VCs) containing protected attributes after off-chain attribute verification.

For service access, users generate zero-knowledge proofs (ZKPs) demonstrating predicate satisfaction (e.g., age $\geq$ 18 without revealing exact birthdates). SPs verify these ZKPs against blockchain-stored DID records and credential status, learning only predicate validity. The blockchain serves as a decentralized ledger for DID documents and VC metadata.

For credential lifecycle management, IdPs may revoke compromised credentials via signed revocation transactions, while users initiate updates by submitting new attribute commitments with ZKPs of validity. These operations follow standard DID protocols for revocation and updates, ensuring compatibility with existing ecosystems.

The system's core operations (excluding revocation/update) are implemented through the following algorithms:

1) $\mathsf{Setup}(1^\lambda) \to pp$: Setup is executed by some party. Takes a security parameter λ, Setup generates public parameters pp and publishes them, which are available to all parties. The setup is done only once.
2) $\mathsf{CreateDIDD}(pp) \to (didd, pk, sk)$: This is executed by users. Take pp as input, CreateDIDD generates key pair(pk, sk) and DID document $didd$ containing pk.
3) $\mathsf{ApplyCred}(pp, v, r) \to (C, \pi_r)$: With value v corresponding to attribute $attr$, random number r and pp as input, user executes ApplyCred to get protected attribute value C and ZKP π_r.
4) $\mathsf{GenCred}(pp, didd, attr, v, C, \pi_r, \mathsf{Proof}_v) \to VC$: With $pp, didd, attr, v, C, \pi_r$ and Proof_v (valid external evidence for v) as input, IdP performs GenCred to generate verifiable credential VC.
5) $\mathsf{ApplyServ}(pp, v, r, \theta) \to \pi_\theta$: With pp, v, r and predicate θ specifying a condition on v (e.g., $\theta :=$ "$v \geq 18$" for

an age attribute) as input, user performs $\mathsf{ApplyServ}$ to generate ZKP π_θ.

6) $\mathsf{VerifyServ}(pp, C, \theta, \pi_\theta) \to \{0,1\}$: With $pp, C, \theta, \pi_\theta$ as input, SP executes $\mathsf{VerifyServ}$ to determine whether to grant the service or not.

C. Definitions

Definition 1(Completeness) A DIMSEPP scheme $\Pi = (\mathsf{Setup}, \mathsf{CreateDIDD}, \mathsf{ApplyCred}, \mathsf{GenCred}, \mathsf{ApplyServ}, \mathsf{VerifyServ})$ is *complete* if for any polynomial-size ledger sampler $\mathcal{S}$ and security parameter λ, the following incompleteness advantages are negligible in λ:

- **VC Generation Completeness**: $\mathsf{Adv}^{\mathsf{VC\text{-}COMP}}_{\Pi,\mathcal{S}} = \Pr[\mathsf{VC\text{-}COMP}(\Pi, \mathcal{S}, \lambda) = 1]$
- **Service Access Completeness**: $\mathsf{Adv}^{\mathsf{SERV\text{-}COMP}}_{\Pi,\mathcal{S}} = \Pr[\mathsf{SERV\text{-}COMP}(\Pi, \mathcal{S}, \lambda) = 1]$

The experiments proceed as follows:

- $\mathsf{VC\text{-}COMP}(\Pi, \mathcal{S}, \lambda)$:
  1) Challenger $\mathcal{C}$ generates $pp \leftarrow \mathsf{Setup}(1^\lambda)$ and sends to $\mathcal{S}$
  2) $\mathcal{S}$ submits to $\mathcal{C}$:
     - Ledger state L containing $didd$s and VCs
     - Attribute tuple $(attr, v, r, \mathsf{Proof}_v)$ where Proof_v is valid external evidence for v
  3) $\mathcal{C}$ generates $(C, \pi_r) \leftarrow \mathsf{ApplyCred}(pp, v, r)$
  4) $\mathcal{C}$ outputs 1 if $\mathsf{GenCred}(pp, didd, attr, v, C, \pi_r, \mathsf{Proof}_v)$ fails to produce valid VC; else 0
- $\mathsf{SERV\text{-}COMP}(\Pi, \mathcal{S}, \lambda)$:
  1) $\mathcal{C}$ initializes as in $\mathsf{VC\text{-}COMP}$
  2) $\mathcal{S}$ submits to $\mathcal{C}$:
     - L containing $didd$s and VCs (containing C)
     - Tuple (v, r, θ), where v, r is contained in C and θ is a specified predicate (e.g., $\geq \theta.min$)
  3) $\mathcal{C}$ generates $\pi_\theta \leftarrow \mathsf{ApplyServ}(pp, v, r, \theta)$
  4) $\mathcal{C}$ outputs 1 if either:
     - $\mathsf{VerifyServ}(pp, C, \theta, \pi_\theta) = 0$ when v satisfies θ, or
     - $\mathsf{VerifyServ}(pp, C, \theta, \pi_\theta) = 1$ when v violates θ

     Else outputs 0

Definition 2 (Security) A DIMSEPP scheme Π is *secure* if it satisfies *consistency*, *attribute-based indistinguishability* and *predicate-based indistinguishability*.

Each of the properties is defined by an experiment involving a DIMSEPP oracle $\mathcal{O}^{\mathsf{DIMSEPP}}$ that receives and answers queries from an adversary $\mathcal{A}$. The queries are proxied via $\mathcal{C}$ that performs the experiment-specifc sanity checks.

$\mathcal{O}^{\mathsf{DIMSEPP}}$ maintains state $(pp, L, \mathsf{KEYS}, \mathsf{VR})$, initialized with $pp \leftarrow \mathsf{Setup}(1^\lambda)$ and empty sets $L, \mathsf{KEYS}, \mathsf{VR}$. Here, L is designed to be **publicly readable** by all parties. The oracle processes the following query types:

- $Q = (\mathsf{CreateDIDD})$:
  - Compute $(didd, pk, sk) \leftarrow \mathsf{CreateDIDD}(pp)$
  - Update $\mathsf{KEYS} \leftarrow \mathsf{KEYS} \cup \{(pk, sk)\}$
  - Update $L \leftarrow L \cup didd$
- $Q = (\mathsf{ApplyCred}, didd, attr, v)$:
  - Choose random $r \xleftarrow{\$} \mathbb{Z}_q$
  - Compute $(C, \pi_r) \leftarrow \mathsf{ApplyCred}(pp, v, r)$
  - Update $\mathsf{VR} \leftarrow \mathsf{VR} \cup (didd, attr, v, r, C)$
  - Return (C, π_r)
- $Q = (\mathsf{GenCred}, didd, attr, v, C, \pi_r, \mathsf{Proof}_v)$:
  - Verify $didd \in L$; if fails, abort.
  - If $\mathsf{Verify}(C, v, \pi_r) = 1 \wedge \mathsf{VerifyEvidence}(\mathsf{Proof}_v) = 1$:
    - $VC \leftarrow \mathsf{GenCred}(pp, didd, attr, v, C, \pi_r, \mathsf{Proof}_v)$
    - Update $L \leftarrow L \cup \{VC\}$
- $Q = (\mathsf{ApplyServ}, didd, \theta)$:
  - Verify $didd \in L$; if fails, abort
  - Retrieve (v, r, C) associated with $(didd, \theta.attr)$ from VR; if fails, abort
  - If v doesn't satisfy θ, abort
  - Compute $\pi_\theta \leftarrow \mathrm{ApplyServ}(pp, v, r, \theta)$.
  - Return π_θ (does not update state).
- $Q = (\mathsf{VerifyServ}, didd, \theta, \pi_\theta)$:
  - Verify $didd \in L$; if fails, abort.
  - Retrieve C from $L.VC$ associated with $(didd, \theta.attr)$; if fails, abort.
  - Compute $b \leftarrow \mathrm{VerifyServ}(pp, C, \theta, \pi_\theta)$.
  - Return b (does not update state).

Definition 2.1 (Consistency) A DIMSEPP scheme Π satisfies *consistency* if for all Probabilistic Polynomial-Time (PPT) adversaries $\mathcal{A}$ and security parameter λ, the advantage $\mathsf{Adv}^{\mathsf{CONS}}_{\Pi,\mathcal{A}}(\lambda)$ is negligible:

$$\mathsf{Adv}^{\mathsf{CONS}}_{\Pi,\mathcal{A}}(\lambda) := \Pr[\mathsf{CONS}(\Pi, \mathcal{A}, \lambda) = 1]$$

The experiment $\mathsf{CONS}(\Pi, \mathcal{A}, \lambda)$ proceeds as:

1) **Initialization:**
   - $\mathcal{C}$ initializes $\mathcal{O}^{\mathsf{DIMSEPP}}$ and gives pp to $\mathcal{A}$
2) **Query Phase:**
   - $\mathcal{A}$ makes adaptive queries to $\mathcal{O}^{\mathsf{DIMSEPP}}$
3) **Challenge Phase:**
   - $\mathcal{A}$ outputs $(v^*, v'^*, r^*, r'^*, C^*, \pi^*_r)$ with:
     - $C^* := [v'^*]G + [r'^*]H$ is a valid Pedersen commitment to (v'^*, r'^*) (see III-A)
     - $\mathsf{Schnorr.Verify}(C^* - [v^*]G,\ H,\ \pi^*_r) = 1$ (see III-A)
     - $[v'^* - v^*]G \neq \mathcal{O}$
4) **Output Phase:**
   - $\mathcal{C}$ outputs 0 unless all of the following hold:
     - $\mathsf{Schnorr.Verify}(C^* - [v^*]G, H, \pi^*_r) = 1$
     - $[v'^* - v^*]G \neq \mathcal{O}$
   - $\mathcal{C}$ attempts $\mathsf{GenCred}(pp, didd, attr, v^*, C^*, \pi^*_r, \mathsf{Proof}^*_v)$ and outputs 1 if valid VC^* is generated, else 0

Definition 2.2 (Attribute-based Indistinguishability) A DIMSEPP scheme Π achieves *attribute-based indistinguishability* if for PPT adversaries $\mathcal{A}$ and security parameter λ, the advantage $\mathsf{Adv}^{\mathsf{AIND}}_{\Pi,\mathcal{A}}(\lambda)$ is negligible:

$$\mathsf{Adv}^{\mathsf{AIND}}_{\Pi,\mathcal{A}}(\lambda) := |\Pr[\mathsf{AIND}(\Pi, \mathcal{A}, \lambda) = 1] - \frac{1}{2}|$$

The experiment $\mathsf{AIND}(\Pi, \mathcal{A}, \lambda)$ proceeds as:

1) **Initialization:**
   - $\mathcal{C}$ generates $pp \leftarrow \mathrm{Setup}(1^\lambda)$
   - $\mathcal{C}$ initializes $\mathcal{O}^{\mathrm{DIMSEPP}}$ with pp
   - $\mathcal{C}$ provides pp to $\mathcal{A}$ and oracle access to $\mathcal{O}^{\mathrm{DIMSEPP}}$
2) **Query Phase 1:**
   - $\mathcal{A}$ adaptively queries $\mathcal{O}^{\mathrm{DIMSEPP}}$ through $\mathcal{C}$ for:
     - DID creation: $(didd, pk, sk) \leftarrow \mathrm{CreateDIDD}(pp)$
     - Credential application: $(C, \pi_r) \leftarrow \mathrm{ApplyCred}(pp, v, r)$
     - Credential generation: $VC \leftarrow \mathrm{GenCred}(pp, didd, attr, v, C, \pi_r, \mathrm{Proof}_v)$
3) **Challenge Phase:**
   - $\mathcal{A}$ submits (v_0, v_1) where $v_0 \neq v_1$
   - $\mathcal{C}$ selects $b \leftarrow \{0, 1\}$
   - $\mathcal{C}$ invokes $\mathcal{O}^{\mathrm{DIMSEPP}}$ to compute $C_b := [v_b]G + [r_b]H$ with $r_b \leftarrow \mathbb{Z}_q$
   - $\mathcal{C}$ returns C_b to $\mathcal{A}$
4) **Query Phase 2:**
   - $\mathcal{A}$ continues queries to $\mathcal{O}^{\mathrm{DIMSEPP}}$ through $\mathcal{C}$
   - **Restriction:** No direct queries about (v_b, C_b)
5) **Output Phase:**
   - $\mathcal{A}$ outputs $b^* \in \{0, 1\}$ as its guess to b
   - $\mathcal{C}$ outputs 1 if $b = b^*$, else 0

The DIMSEPP oracle $\mathcal{O}^{\mathrm{DIMSEPP}}$ maintains all system state (including ledger L, key store KEYS, and credential records VR), while $\mathcal{C}$ acts as a relay between $\mathcal{A}$ and $\mathcal{O}^{\mathrm{DIMSEPP}}$. All cryptographic computations are performed by $\mathcal{O}^{\mathrm{DIMSEPP}}$ to properly model system execution.

Definition 2.3 (Predicate-Based Indistinguishability) A DIMSEPP scheme Π satisfies *predicate-based indistinguishability* if for all PPT adversaries $\mathcal{A}$ and security parameter λ, the advantage $\mathrm{Adv}^{\mathrm{PIND}}_{\Pi,\mathcal{A}}(\lambda)$ is negligible, where the experiment $\mathrm{PIND}(\Pi, \mathcal{A}, \lambda)$ proceeds as:

1) **Initialization:**
   - $\mathcal{C}$ generates $pp \leftarrow \mathsf{Setup}(1^\lambda)$
   - $\mathcal{C}$ initializes $\mathcal{O}^{\mathsf{DIMSEPP}}$ with pp
   - $\mathcal{C}$ provides pp to $\mathcal{A}$ and oracle access to $\mathcal{O}^{\mathsf{DIMSEPP}}$
2) **Query Phase 1:**
   - $\mathcal{A}$ adaptively queries $\mathcal{O}^{\mathsf{DIMSEPP}}$ through $\mathcal{C}$ for:
     - DID creation: $(didd, pk, sk) \leftarrow \mathsf{CreateDIDD}(pp)$
     - Credential application: $(C, \pi_r) \leftarrow \mathsf{ApplyCred}(pp, v, r)$
     - Credential generation: $VC \leftarrow \mathsf{GenCred}(pp, didd, attr, v, C, \pi_r, \mathsf{Proof}_v)$
     - Service access: $\pi_\theta \leftarrow \mathsf{ApplyServ}(pp, v, r, \theta)$
     - Service verification: $b \leftarrow \mathrm{VerifyServ}(pp, C, \theta, \pi_\theta)$
3) **Challenge Phase:**
   - $\mathcal{A}$ submits (v_0, v_1, θ) where:
     - $v_0 \neq v_1$
     - Both v_0 and v_1 satisfy the θ
   - $\mathcal{C}$ selects $b \leftarrow \{0, 1\}$
   - $\mathcal{C}$ invokes $\mathcal{O}^{\mathsf{DIMSEPP}}$ to compute:

$$C_b := [v_b]G + [r_b]H \text{ with } r_b \leftarrow \mathbb{Z}_q$$
$$\pi_\theta := \mathsf{ApplyServ}(pp, v_b, r_b, \theta)$$

   - $\mathcal{C}$ returns (C_b, π_θ) to $\mathcal{A}$
4) **Query Phase 2:**
   - $\mathcal{A}$ continues queries to $\mathcal{O}^{\mathsf{DIMSEPP}}$ through $\mathcal{C}$
   - **Restriction:** No queries about (v_b, C_b, π_θ)
5) **Output Phase:**
   - $\mathcal{A}$ outputs $b^* \in \{0, 1\}$ as its guess for b
   - $\mathcal{C}$ outputs 1 if $b = b^*$, else 0

The advantage is defined as:

$$\mathrm{Adv}^{\mathrm{PIND}}_{\Pi,\mathcal{A}}(\lambda) := \left| \Pr[\mathrm{PIND}(\Pi, \mathcal{A}, \lambda) = 1] - \frac{1}{2} \right|$$

III. Decentralized Identity Management System with Enhanced Privacy Protection(DIMSEPP)

A. Mathematical Background

1) Pedersen commitment: Let $\mathbb{G}$ be a cyclic subgroup of an elliptic curve $E(\mathbb{F}_p)$ of prime order q, with base points $G, H \in \mathbb{G}$ where H is a random point with unknown discrete log with respect to G. The commitment [26] to a secret value $v \in \mathbb{Z}_q$ with random number $r \in \mathbb{Z}_q$ is:

$$C(v, r) = [v]G + [r]H \tag{1}$$

where $[n]P$ denotes scalar multiplication on $E(\mathbb{F}_p)$.

Key Properties

- **Perfect Hiding** For any $v' \in \mathbb{Z}_q$, there exists $r' \in \mathbb{Z}_q$ such that $C(v, r) = C(v', r')$. Thus, $C(v, r)$ reveals no information about v.
- **Computational Binding** Under the **Elliptic Curve Discrete Logarithm Problem (ECDLP)** assumption, no PPT adversary can find $(v', r') \neq (v, r)$ satisfying:

$$[v]G + [r]H = [v']G + [r']H. \tag{2}$$

2) Non-Interactive Schnorr Proof (Schnorr): Let $\mathbb{G}$ be a cyclic subgroup of an elliptic curve $E(\mathbb{F}_p)$ with prime order q and generator G. The non-interactive Schnorr proof[32], [33] allows a prover $\mathcal{P}$ to demonstrate knowledge of a secret $x \in \mathbb{Z}_q$ for a public key $Y = [x]G$ without interaction. This mechanism involves two cryptographic operations: Prove (executed by $\mathcal{P}$ to generate proofs) and Verify (performed by verifier $\mathcal{V}$ to validate proofs without accessing sensitive data). $\mathrm{Prove}(Y, G, x) \to \pi$:

1) $\mathcal{P}$ selects $r \xleftarrow{\$} \mathbb{Z}_q$ and computes $R = [r]G$.
2) Computes $c = \mathcal{H}(R \parallel Y)$, where $\mathcal{H} : \{0, 1\}^* \to \mathbb{Z}_q$ is a cryptographic hash function.
3) Computes $s = (r + c \cdot x) \bmod q$.
4) Outputs the proof $\pi = (R, s)$.

$\mathrm{Verify}(Y, G, \pi) \to b, b \in \{0, 1\}$:

1) $\mathcal{V}$ recomputes $c = \mathcal{H}(R \parallel Y)$.
2) if $[s]G \stackrel{?}{=} R + [c]Y$, outputs 1; else outputs 0.

Key Properties:

- **Unforgeability:** Secure against existential forgery under chosen-message attacks (EUF-CMA) in the random oracle model, assuming the hardness of the discrete logarithm problem in $\mathbb{G}$.
- **Zero-Knowledge:** The protocol is simulatable via:
  1) Choose random $c^* \xleftarrow{\$} \mathbb{Z}_q$ and $s^* \xleftarrow{\$} \mathbb{Z}_q$
  2) Compute $R^* = [s^*]G - [c^*]Y$
  3) Program the random oracle to satisfy $\mathcal{H}(R^* \parallel Y) := c^*$

  The simulation is perfect since (R^*, s^*) is identically distributed to real transcripts.
- **Size:** The proof π consists of one curve point (R) and one scalar (s), totaling $\lceil \log_2 q \rceil + |\mathbb{G}|$ bits.

3) Ring Signatures (RingSig): A ring signature scheme[34], [35], [36] allows a signer to anonymously sign a message on behalf of a *ring* (set) of public keys. Formally, given:

- A ring of n public keys: $R_{PK} = \{PK_1, \ldots, PK_n\}$
- Signer's secret key sk_s corresponding to $PK_s \in R_{PK}$
- Message $m \in \{0,1\}^*$

The scheme consists of algorithms:

$$\sigma \leftarrow \mathsf{Sign}(sk_s, R_{PK}, m) \tag{3}$$

$$\{0,1\} \leftarrow \mathsf{Verify}(R_{PK}, m, \sigma) \tag{4}$$

Key Properties:

- **Anonymity**: For any $PK_i \in R_{PK}$, the probability that an adversary identifies s is at most $1/n + \mathsf{negl}(\lambda)$.
- **Unforgeability**: No PPT adversary can produce a valid σ without knowing any sk_i for $PK_i \in R_{PK}$, under the Discrete Logarithm assumption.

4) Ring Signature-based Range Proof (RingProof): The scheme[31] enables a prover to demonstrate that a committed value v lies within the range $[0, 2^n - 1]$ without revealing v or the random number r. This is achieved by combining Pedersen commitments with ring signatures for each bit of v.

The scheme consists of two main phases: Prove and Verify.

- $\mathsf{Prove}(v, r) \to \pi$, where $v \in [0, 2^n - 1]$, $r \in \mathbb{Z}_q$, $C = [v]G + [r]H$:
  1) Bit decomposition and random number allocation:
     - Represent v in binary: $v = \sum_{j=0}^{n-1} b_j 2^j$ where $b_j \in \{0,1\}$ for $j \in \{0, 1, \ldots, n-1\}$
     - For $j \in \{0, 1, \ldots, n-2\}$, sample $r_j \xleftarrow{\$} \mathbb{Z}_q$
     - Compute $r_{n-1} = r - \sum_{j=0}^{n-2} r_j \bmod q$
  2) Ring construction for each bit:

$$PK_{j,0} = [r_j]H + [b_j 2^j]G \tag{5}$$

$$PK_{j,1} = PK_{j,0} - [2^j]G \tag{6}$$

$$R_j = \{PK_{j,0}, PK_{j,1}\} \tag{7}$$

  3) Ring signature generation:

$$\sigma_j \leftarrow \mathsf{RingSig.Sign}(r_j, R_j, PK_{j,0}) \tag{8}$$

  where r_j is the secret key for PK_{j,b_j}, $PK_{j,0}$ is used as message to sign. Return $\pi = \{\sigma_j\}_{j=0}^{n-1}$

- $\mathsf{Verify}(C, \pi) \to b$, where $b \in \{0,1\}$:
  1) Verify all ring signatures: $\mathsf{RingSig.Verify}(R_j, PK_{j,0}, \sigma_j) = 1$ for all $j \in \{0, 1, \ldots, n-1\}$; else return 0.
  2) Check commitment consistency: $\sum_{j=0}^{n-1} PK_{j,0} \stackrel{?}{=} C$; else return 0.
  3) Return 1 if all checks pass.

Key Properties:

- **Perfect Privacy**: The proof reveals nothing beyond $v \in [0, 2^n - 1]$
- **Computational Soundness**: Forging a proof requires solving the ECDLP or breaking the ring signature scheme
- **Compactness**: Proof size is $O(n)$

B. The Proposed Scheme

The system architecture, comprising six algorithms, partitions into four operational phases: Initial, Registration, Credential Issuance and Service Access.

1) Initial Phase: The Setup algorithm is executed by some participant to generate public parameters. To ensure the discrete logarithm relationship between generators G and H remains unknown, a hash-to-curve technique can be employed.
$\mathsf{Setup}(1^\lambda) \to pp$: Outputs public parameters pp including:

- Elliptic curve group $\mathbb{G}$ with generators (G, H)
- Cryptographic hash functions $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_q$

2) Registration Phase: The registration phase follows standard DID operations.
$\mathsf{CreateDIDD}(pp) \to (didd, pk, sk)$

1) Generate $(sk, pk) = (d, [d]G)$,
2) Construct DID document $\mathsf{didd} = \{\texttt{id} : \mathsf{did}, \texttt{verificationMethod} : [pk], \texttt{service} : \ldots\}$
3) Return $(didd, pk, sk)$

User executes CreateDIDD and obtains $(didd, pk, sk)$, then stores (pk, sk) locally and submits didd to the blockchain system. The blockchain system verifies the didd's integrity and stores it on-chain after consensus validation.

3) Credential Issuance Phase: When there is a need to apply VC corresponding to $(attr, v)$, user chooses r via $r \xleftarrow{\$} \mathbb{Z}_q$, executes $\mathsf{ApplyCred}$ and obtains (C, π_r), then stores $(didd, attr, v, r, C)$ locally and submits $(didd, attr, v, C, \pi_r, \mathsf{Proof}_v)$ to IdP, where Proof_v is a formally verifiable evidence of the attribute
$\mathsf{ApplyCred}(pp, v, r) \to (C, \pi_r)$

1) Compute Pedersen commitment: $C = [v]G + [r]H$
2) Produce Schnorr ZKP: $\pi_r \leftarrow \mathsf{Schnorr.Prove}(C - [v]G, H, r)$
3) Return (C, π_r)

Example for birth date:

- Let $attr =$ birth_date, $v = 19970901$
- Proof_v includes:
  - Complete ePassport Data Structure EDS
  - EDS digital signature σ_{EDS}

On receiving $(didd, attr, v, C, \pi_r, \mathsf{Proof}_v)$ from the user, the IdP first verifies that $didd$ is a valid and unrevoked

DID document registered on the blockchain. It then executes $\mathsf{GenCred}$ to generate VC and submits it to the blockchain system.

$\mathsf{GenCred}(pp, didd, attr, v, C, \pi_r, \mathsf{Proof}_v) \rightarrow VC$

1) Check $\mathsf{VerifyEvidence}(\mathsf{Proof}_v) = 1$; if not, return $\perp$ (*For birth date:* Check EDS and σ_{EDS})
2) Check $\mathsf{Schnorr.Verify}(C - [v]G, H, \pi_{\text{r}}) = 1$; if not, return $\perp$
3) Generate $VC = \{\texttt{id} : didd.id, \texttt{type} : attr, \texttt{value} : C, \texttt{issuerSig} : \sigma_{\text{issuer}}, \dots\}$
4) Return VC

4) Service Access Phase: The service access predicate θ in the system supports three canonical forms: (i) lower-bound assertions ($v \geq \theta.min$), (ii) upper-bound constraints ($v \leq \theta.max$), and (iii) equality conditions ($v = \theta.v$). When need to access service, the user first retrieves $(didd, \theta.attr, v, r, C)$ locally via $(didd, \theta.attr)$, then verifies whether its v satisfies θ. If compliant, the user, executes $\mathsf{ApplyServ}$ and obtains π_θ. Then, it submits $(didd, \theta, \pi_\theta)$ to SP.

Notably, all proofs $\pi_\theta = \{\sigma_j\}_{j=0}^{n-1}$ must use a fixed proof size n_{max} determined by system parameters, with zero-padding applied when necessary, to prevent size-based distinguishability.

$\mathsf{ApplyServ}(pp, v, r, \theta) \rightarrow \pi_\theta$

1) If θ is of the form $v \geq \theta.min$:
   - Generate the range proof $\pi_\theta := \mathsf{RingProof.Prove}(v - \theta.min, r)$
   - Return π_θ
2) If θ is of the form $v \leq \theta.max$:
   - Generate the range proof $\pi_\theta := \mathsf{RingProof.Prove}(\theta.max - v, -r)$
   - Return π_θ
3) If θ is of the form $v = \theta.v$:
   - Generate the schnorr proof $\pi_\theta := \mathsf{Schnorr.Prove}(rH, H, r)$
   - Return π_θ

On receiving application from user, SP retrieves $C := VC.C$ via $(didd, \theta.attr)$ from blockchain system. Then, SP executes $\mathsf{VerifyServ}$ and obtains b_θ. If $b_\theta = 1$, SP grants access; else rejects.

$\mathsf{VerifyServ}(pp, C, \theta, \pi_\theta) \rightarrow b_\theta$

1) If θ is of the form $v \geq \theta.min$:
   - Compute $b_\theta = \mathsf{RingProof.Verify}(C - [\theta.min]G, \pi_\theta)$
   - Return b_θ
2) If θ is of the form $v \leq \theta.max$:
   - Compute $b_\theta = \mathsf{RingProof.Verify}([\theta.max]G - C, \pi_\theta)$
   - Return b_θ
3) If θ is of the form $v = \theta.v$:
   - Compute $b_\theta = \mathsf{Schnorr.Verify}(C - [\theta.v]G, H, \pi_\theta)$
   - Return b_θ

For constraints requiring both conditions ($\theta.min \leq v \leq \theta.max$), the user must provide both proofs and the SP will grant access iff both proofs are verified.

IV. SECURITY ANALYSIS

A. Proof of Completeness

Theorem 1. The DIMSEPP scheme satisfies *completeness* (Definition 1): For any PPT ledger sampler $\mathcal{S}$ and $\lambda \in \mathbb{N}$, the incompleteness advantage $\text{Adv}^{\text{VC-COMP}}_{\Pi,\mathcal{S}}$ and $\text{Adv}^{\text{SERV-COMP}}_{\Pi,\mathcal{S}}$ are negligible in λ.

Proof. We analyze both experiments in Definition 1.

1. VC Generation Completeness:

- $\mathsf{ApplyCred}$ outputs (C, π_r) where $C = [v]G + [r]H$ and π_r is a valid Schnorr proof for r.
- $\mathsf{GenCred}$ succeeds if Proof_v and π_r verify (which they do by construction).
- Thus, $\text{Adv}^{\text{VC-COMP}}_{\Pi,\mathcal{S}} = \text{negl}(\lambda)$.

2. Service Access Completeness:

- For $\theta := (v \geq \theta.\text{min})$ or $\theta := (v \leq \theta.\text{max})$, $\mathsf{ApplyServ}$ generates a range proof π_θ for $v - \theta.\text{min} \in [0, 2^n - 1]$ or $\theta.\text{max} - v \in [0, 2^n - 1]$, which verifies by RingProof completeness.
- For $\theta := (v = \theta.v)$, the Schnorr proof π_θ verifies by Schnorr completeness.
- Thus, $\text{Adv}^{\text{SERV-COMP}}_{\Pi,\mathcal{S}} = \text{negl}(\lambda)$.

□

B. Proof of Consistency

Theorem 2. Under the *Elliptic Curve Discrete Logarithm Problem (ECDLP)* assumption, the DIMSEPP scheme satisfies *consistency* as defined in Definition 2.1.

Proof. We prove consistency by contradiction. Assume there exists a PPT adversary $\mathcal{A}$ that breaks consistency with non-negligible advantage $\text{Adv}^{\text{CONS}}_{\Pi,\mathcal{A}}(\lambda)$. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the ECDLP.

a) Reduction to ECDLP:

1) $\mathcal{B}$ receives an ECDLP instance (G, H), where $H = [x]G$ for unknown $x \in \mathbb{Z}_q$.
2) $\mathcal{B}$ simulates the DIMSEPP environment for $\mathcal{A}$:
   - Generates public parameters $pp = (\mathbb{G}, G, H, \mathcal{H})$.
   - Provides $\mathcal{A}$ with pp and oracle access to $\mathcal{O}^{\text{DIMSEPP}}$.
3) $\mathcal{A}$ outputs $(v^*, v'^*, r^*, r'^*, C^*, \pi_r^*)$ such that: $C^* = [v'^*]G + [r'^*]H$, and π_r^* verifies $C^* - [v^*]G = [r^*]H$.
4) From the verification equation:

$$[v'^* - v^*]G = [r^* - r'^*]H.$$

Since $v'^* \neq v^*$, $\mathcal{B}$ computes:

$$x = \frac{v'^* - v^*}{r^* - r'^*} \mod q,$$

solving the ECDLP for $H = [x]G$.

b) Contradiction: If $\mathcal{A}$ succeeds with non-negligible probability, $\mathcal{B}$ solves ECDLP with the same probability. However, ECDLP is assumed to be hard, so $\text{Adv}^{\text{CONS}}_{\Pi,\mathcal{A}}(\lambda)$ must be negligible.

c) Conclusion: DIMSEPP satisfies consistency under the ECDLP assumption. □

C. Proof of Attribute-Based Indistinguishability

Theorem 3. The DIMSEPP scheme satisfies *attribute-based indistinguishability* (Definition 2.2) under the computational hiding property of Pedersen commitments.

Proof. We prove attribute-based indistinguishability by constructing a reduction to the hiding property of Pedersen commitments. Assume there exists a PPT adversary $\mathcal{A}$ that breaks attribute-based indistinguishability with non-negligible advantage $\text{Adv}^{\text{AIND}}_{\Pi,\mathcal{A}}(\lambda)$. We build an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to attack Pedersen's hiding property.

a) Reduction to Pedersen Hiding:

1) $\mathcal{B}$ receives a Pedersen commitment challenge (G, H) where $H = [x]G$ for unknown $x \in \mathbb{Z}_q$.
2) $\mathcal{B}$ simulates the DIMSEPP environment for $\mathcal{A}$:
   - Generates public parameters $pp = (\mathbb{G}, G, H, \mathcal{H})$.
   - Provides $\mathcal{A}$ with pp and oracle access to $\mathcal{O}^{\text{DIMSEPP}}$.
3) In the challenge phase:
   - $\mathcal{A}$ submits (v_0, v_1) with $v_0 \neq v_1$.
   - $\mathcal{B}$ forwards (v_0, v_1) to the Pedersen challenger and receives $C_b = [v_b]G + [r]H$ for random $b \leftarrow \{0, 1\}$.
   - $\mathcal{B}$ gives C_b to $\mathcal{A}$.
4) After additional queries (excluding direct queries about C_b), $\mathcal{A}$ outputs a guess b^*.
5) $\mathcal{B}$ outputs b^* as its solution to the Pedersen challenge.

b) Analysis:

- The simulation is perfect because:
  - All oracle queries are answered using real DIMSEPP operations.
  - C_b is distributed identically to a real DIMSEPP commitment.
- $\mathcal{A}$'s advantage is preserved:

$$\left|\Pr[\mathcal{B} \text{ wins}] - \frac{1}{2}\right| = \text{Adv}^{\text{AIND}}_{\Pi,\mathcal{A}}(\lambda)$$

- If $\mathcal{A}$ succeeds with non-negligible advantage, $\mathcal{B}$ breaks Pedersen's hiding property with the same advantage.

c) Contradiction: Pedersen commitment is perfect hiding. Therefore, $\text{Adv}^{\text{AIND}}_{\Pi,\mathcal{A}}(\lambda)$ must be negligible.

d) Conclusion: DIMSEPP satisfies attribute-based indistinguishability under the hiding property of Pedersen commitment. □

D. Proof of Predicate-Based Indistinguishability

Theorem 4. The DIMSEPP scheme satisfies *predicate-based indistinguishability* (Definition 2.3) under the ring signature anonymity.

Proof. We prove predicate-based indistinguishability by constructing a reduction to the anonymity of the underlying ring signature scheme. Assume there exists a PPT adversary $\mathcal{A}$ that breaks the indistinguishability with non-negligible advantage $\text{Adv}^{\text{PIND}}_{\Pi,\mathcal{A}}(\lambda)$. We prove that $\mathcal{A}$ could attack the anonymity of the ring signature.

a) Reduction to Ring Signature Anonymity:

1) $\mathcal{C}$ simulates the DIMSEPP environment for $\mathcal{A}$:
   - Generates $pp = (\mathbb{G}, G, H, \mathcal{H})$ and initializes $\mathcal{O}^{\text{DIMSEPP}}$.
   - Provides $\mathcal{A}$ with pp and oracle access to $\mathcal{O}^{\text{DIMSEPP}}$.
2) In the challenge phase:
   - $\mathcal{A}$ submits (v_0, v_1, θ) where $v_0 \neq v_1$ and both satisfy θ.
   - $\mathcal{C}$ obtains and transmits (C_b, π_θ) to $\mathcal{A}$ as defined in Definition 2.3.
3) $\mathcal{A}$ continues queries under the standard restrictions.
4) $\mathcal{A}$ outputs b^*, then computes $v'_{b^*} = v_{b^*} - \theta.min$ or $v'_{b^*} = \theta.max - v_{b^*}$ depending on the form of θ.
5) Let $\text{Bit}(x, j)$ denote the j-th bit of x. $\forall j \in \{0, 1, ...n-1\}$, $\mathcal{A}$ outputs $\text{Bit}(v'_{b^*}, j)$ as its guess for $s \in \{0, 1\}$ in σ_j of π_θ, which represents the index of the private key used to generate σ_j.

b) Analysis:

- The simulation is perfect because all non-challenge operations are honestly executed.
- $\mathcal{A}$'s advantage is preserved:

$$\text{Adv}^{\text{RS}-\text{ANON}}_{\mathcal{A}}(\lambda) = \text{Adv}^{\text{PIND}}_{\Pi,\mathcal{A}}(\lambda)$$

  Here, $\text{Adv}^{\text{RS}-\text{ANON}}_{\mathcal{A}}(\lambda)$ denotes the advantage of $\mathcal{A}$ in breaking the anonymity of the ring signature scheme.
- If $\mathcal{A}$ distinguishes between v_0 and v_1, it breaks ring signature anonymity.

c) Contradiction: The ring signature scheme is unconditionally anonymous. Therefore, $\text{Adv}^{\text{PIND}}_{\Pi,\mathcal{A}}(\lambda)$ must be negligible.

d) Conclusion: DIMSEPP satisfies predicate-based indistinguishability under the anonymity of the underlying ring signature scheme.

□

E. Security Summary

Combining Theorems 2–4, DIMSEPP achieves *provable security* under standard cryptographic assumptions (ECDLP, Pedersen commitment hiding, and ring signature anonymity), satisfying all requirements in Definition 2: consistency, attribute-based and predicate-based indistinguishability.

V. PERFORMANCE ANALYSIS

A. Theoretical Analysis

Our comparative analysis (Table I) evaluates DIMSEPP against three representative systems across four key dimensions that highlight its novel contributions.

Where existing systems force tradeoffs between privacy and functionality—requiring either raw data exposure ([16]), complete attribute disclosure ([17]), or predicate leakage ([18])—DIMSEPP's commitment-based approach uniquely preserves both verifiability and confidentiality. While [17] and [18] rely on zk-SNARKs[37], [38] that require complex trusted setup procedures - a significant drawback in decentralized

TABLE I
COMPARISON OF DECENTRALIZED IDENTITY MANAGEMENT SYSTEMS

	[16]	[17]	[18]	**DIMSEPP**
On-chain Data	Raw	Protected	Predicate	Commitment
Minimal Disclosure	No	No	Yes	Yes
Requires Trusted Setup	No	Yes	Yes	No
Compatibility	Full	Partial	Partial	Full

environments where reliable third parties are scarce - our system combines Pedersen commitments with efficient ring-signature-based proofs that need no trusted initialization ceremony. This approach demonstrates superior computational and storage efficiency.

Crucially, DIMSEPP achieves these advances without compromising the full W3C compatibility essential for adoption. The comparison reveals our architectural advantage: we resolve the core transparency-privacy tension through lightweight cryptography that outperforms SNARK-based alternatives in both trust assumptions and operational efficiency.

B. Experimental Results

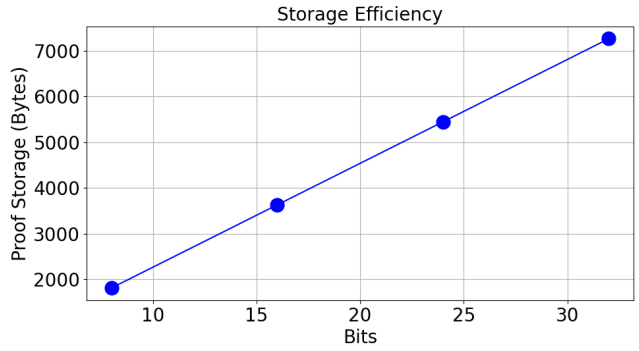

Fig. 2. storage efficiency

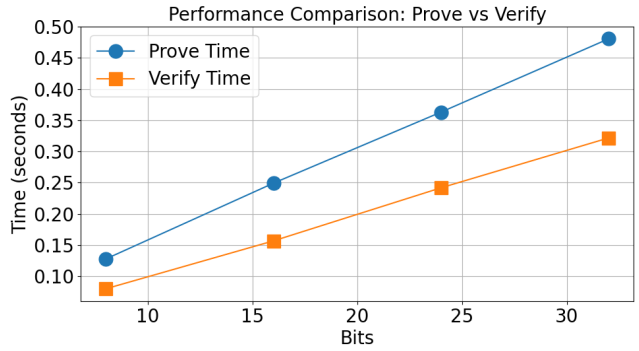

Fig. 3. computational overhead

To validate the practical feasibility of the proposed scheme, particularly its suitability for resource-constrained environments, we implement a simulation. The evaluation focuses on two critical aspects: (1) the efficiency of credential issuance and (2) the performance of service access.

All experiments were conducted in a single-threaded environment to isolate baseline performance metrics and eliminate potential concurrency-related overhead. This setup ensures a clear evaluation of the cryptographic operations' intrinsic efficiency. Note that further performance improvements can be achieved through multi-threading or parallel processing, as key operations (e.g., ring signature generation/verification for individual bits) are inherently parallelizable.

The experiments were performed on a HP laptop running Windows 11 Pro (23H2) with an Intel Core i5-10210U processor and 32GB DDR4 RAM, using a VirtualBox 7.0 virtual machine configured with 4 vCPUs and 16GB RAM running Ubuntu 20.04 LTS. The prototype system, built with Python 3.8.10, employs the starkbank-ecdsa library (v2.2.0) [39] for cryptographic operations on the secp256k1 curve. Besides, the ring signature algorithm from Monero [40] is employed, modified by omitting the components related to R_i, since traceability is not required. The system consistently employs 33-byte compressed elliptic curve points, a key optimization that significantly reduces storage overhead.

For the credential issuance phase, the system demonstrates exceptional efficiency in proof generation and verification. User-side operations produce compact 65-byte Schnorr proofs in just 0.002656 seconds, while IdP verification completes in 0.005045 seconds.

For the service access phase, the system demonstrates promising results across all predicate types. For equality predicates ($v = \theta.v$), the system reuses the Schnorr proof mechanism, achieving identical efficiency to credential issuance proofs. The performance of range predicates ($v \geq \theta.min$ or $v \leq \theta.max$) is shown in Fig. 2 and Fig. 3. Fig. 2 illustrates the linear relationship between proof size and bit length (i.e., the number of bits used to represent $v - \theta.min$ or $\theta.max - v$), where each bit requires only 227 bytes. This optimization stems from our public key compression technique: since $PK_{j,1}$ can be derived directly from $PK_{j,0}$ as $PK_{j,1} = PK_{j,0} - [2^j]G$, the proof only needs to store $PK_{j,0}$. Fig. 3 demonstrates consistent computational performance, where 32-bit verification completes in 0.322 seconds while maintaining a stable 1.5× speed advantage over proof generation.

Notably, a 32-bit length suffices for most practical scenarios, supporting value differences up to 2^{32}.

The combined results from both experimental phases demonstrate that the proposed scheme meets critical requirements for practical deployment:

- **Storage Efficiency**: 33-byte protected value (C) in VC, 65-byte issuance proofs and 227-byte/bit service access proofs
- **Computational Performance**: Sub-second operations (both proof generation and verification in all two phases)
- **Predictable Scaling**: Linear complexity in both dimensions

VI. Conclusion

The DIMSEPP system presents a novel approach to decentralized identity management by addressing critical privacy limitations inherent in current blockchain-based DID implementations. Through the integration of off-chain attribute storage with on-chain commitments and zero-knowledge proof protocols for minimal-disclosure credential presentations, our system achieves enhanced privacy protection without compromising the core benefits of decentralization.

Theoretical analysis and experimental results demonstrate that DIMSEPP successfully resolves the transparency-privacy conflict in existing DID systems while maintaining practical efficiency. With linear complexity in proof size, generation time and verification time, the system achieves sub-second operational performance, making it practical for resource-constrained environments such as mobile identity management and IoT applications.

Security proofs establish that DIMSEPP satisfies completeness, consistency, attribute-based indistinguishability, and predicate-based indistinguishability under standard cryptographic assumptions. These properties ensure that the system meets both regulatory requirements for data minimization and real-world needs for selective disclosure in sensitive domains like healthcare and financial services.

Future work will focus on optimizing proof aggregation techniques and expanding the system's predicate language to support more complex authorization scenarios. The DIMSEPP framework provides a foundation for building trustworthy decentralized identity systems that align with evolving privacy regulations and user expectations in an increasingly digital world.

References

[1] Avellaneda O, Bachmann A, Barbir A, et al. Decentralized identity: Where did it come from and where is it going?[J]. IEEE Communications Standards Magazine, 2019, 3(4): 10-13.

[2] Dib O, Rababah B. Decentralized identity systems: Architecture, challenges, solutions and future directions[J]. Annals of Emerging Technologies in Computing (AETiC), 2020, 4(5): 19-40.

[3] Fdhila W, Stifter N, Kostal K, et al. Methods for decentralized identities: Evaluation and insights[C]//International Conference on Business Process Management. Cham: Springer International Publishing, 2021: 119-135.

[4] Stockburger L, Kokosioulis G, Mukkamala A, et al. Blockchain-enabled decentralized identity management: The case of self-sovereign identity in public transportation[J]. Blockchain: Research and Applications, 2021, 2(2): 100014.

[5] Luecking M, Fries C, Lamberti R, et al. Decentralized identity and trust management framework for Internet of Things[C]//2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). IEEE, 2020: 1-9.

[6] Zhou L, Diro A, Saini A, et al. Leveraging zero knowledge proofs for blockchain-based identity sharing: A survey of advancements, challenges and opportunities[J]. Journal of Information Security and Applications, 2024, 80: 103678.

[7] Ahmed M R, Islam A K M M, Shatabda S, et al. Blockchain-based identity management system and self-sovereign identity ecosystem: A comprehensive survey[J]. IEEE Access, 2022, 10: 113436-113481.

[8] Zheng Z, Xie S, Dai H N, et al. Blockchain challenges and opportunities: A survey[J]. International journal of web and grid services, 2018, 14(4): 352-375.

[9] Nofer M, Gomber P, Hinz O, et al. Blockchain[J]. Business & information systems engineering, 2017, 59(3): 183-187.

[10] Di Pierro M. What is the blockchain?[J]. Computing in Science & Engineering, 2017, 19(5): 92-95.

[11] Yaga D, Mell P, Roby N, et al. Blockchain technology overview[J]. arXiv preprint arXiv:1906.11078, 2019.

[12] Dinh T T A, Liu R, Zhang M, et al. Untangling blockchain: A data processing view of blockchain systems[J]. IEEE transactions on knowledge and data engineering, 2018, 30(7): 1366-1385.

[13] Guo H, Yu X. A survey on blockchain technology and its security[J]. Blockchain: research and applications, 2022, 3(2): 100067.

[14] Dunphy P, Petitcolas F A P. A first look at identity management schemes on the blockchain[J]. IEEE security & privacy, 2018, 16(4): 20-29.

[15] Kuperberg M. Blockchain-based identity management: A survey from the enterprise and ecosystem perspective[J]. IEEE Transactions on Engineering Management, 2019, 67(4): 1008-1027.

[16] https://www.w3.org/TR/did-core/

[17] Yang X, Li W. A zero-knowledge-proof-based digital identity management scheme in blockchain[J]. Computers & Security, 2020, 99: 102050.

[18] Maram D, Malvai H, Zhang F, et al. Candid: Can-do decentralized identity with legacy compatibility, sybil-resistance, and accountability[C]//2021 IEEE Symposium on Security and Privacy (SP). IEEE, 2021: 1348-1366.

[19] Liu, Y., He, D., Obaidat, M.S., et al. (2021). Blockchain-based identity management systems: A review. *Journal of Network and Computer Applications*, 186, 103079.

[20] Voigt P, Von dem Bussche A. The eu general data protection regulation (gdpr)[J]. A practical guide, 1st ed., Cham: Springer International Publishing, 2017

[21] Zaeem R N, Barber K S. The effect of the GDPR on privacy policies: Recent progress and future promise[J]. ACM Transactions on Management Information Systems (TMIS), 2020, 12(1): 1-20.

[22] Albrecht J P. How the GDPR will change the world[J]. Eur. Data Prot. L. Rev., 2016, 2: 287.

[23] ISO/IEC. (2022). Information security, cybersecurity and privacy protection—Information security management systems—Requirements (ISO/IEC 27001:2022). International Organization for Standardization.

[24] National Institute of Standards and Technology (NIST). (2020). NIST privacy framework: A tool for improving privacy through enterprise risk management (Version 1.0). U.S. Department of Commerce. https://doi.org/10.6028/NIST.CSWP.01162020

[25] Yin J, Xiao Y, Feng J, et al. Didtrust: Privacy-preserving trust management for decentralized identity[J]. IEEE Transactions on Dependable and Secure Computing, 2025,vol. 22, no. 3, pp. 3105-3119

[26] Pedersen T P. Non-interactive and information-theoretic secure verifiable secret sharing[C]//Annual international cryptology conference. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991: 129-140.

[27] Damgård I. Commitment schemes and zero-knowledge protocols[C]//School organized by the European Educational Forum. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998: 63-86.

[28] Goldwasser S., Micali S., Rackoff C. Knowledge complexity of interactive proofs [C]//Proceedings of the seventeenth annual ACM symposium on Theory of computing. 1985: 291-304.

[29] Groth J. On the size of pairing-based non-interactive arguments[C]//Annual international conference on the theory and applications of cryptographic techniques. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016: 305-326.

[30] Sun X, Yu F R, Zhang P, et al. A survey on zero-knowledge proof in blockchain[J]. IEEE network, 2021, 35(4): 198-205.

[31] Noether S, Mackenzie A. Ring confidential transactions[J]. Ledger, 2016, 1: 1-18.

[32] Hao, Feng. Schnorr non-interactive zero-knowledge proof. No. rfc8235. 2017.

[33] Camenisch, J., Kiayias, A., Yung, M. (2009). On the Portability of Generalized Schnorr Proofs. In: Joux, A. (eds) Advances in Cryptology - EUROCRYPT 2009.

[34] Bender A, Katz J, Morselli R. Ring signatures: Stronger definitions, and constructions without random oracles[C]//Theory of Cryptography Conference. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006: 60-79.

[35] Rivest, R.L., Shamir, A., Tauman, Y. (2006). How to Leak a Secret: Theory and Applications of Ring Signatures. In: Goldreich, O., Rosenberg, A.L., Selman, A.L. (eds) Theoretical Computer Science. Lecture Notes in Computer Science, vol 3895. Springer, Berlin, Heidelberg.

[36] Wang L, Zhang G, Ma C. A survey of ring signature[J]. Frontiers of Electrical and Electronic Engineering in China, 2008, 3(1): 10-19.

[37] Groth J, Maller M. Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs[C]//Annual International Cryptology Conference. Cham: Springer International Publishing, 2017: 581-612.
[38] Eberhardt J, Tai S. Zokrates-scalable privacy-preserving off-chain computations[C]//2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). IEEE, 2018: 1084-1091.
[39] https://github.com/starkbank/ecdsa-python
[40] https://www.getmonero.org/resources/research-lab/pubs/cryptonote-whitepaper.pdf