

Article

IoT-SBIdM: A Privacy-Preserving Stateless Blockchain-Based Identity Management for Trustworthy Internet of Things IoT Ecosystems

Eman Alatawi *, Anoud Alhawiti , Doaa Albalawi  and Umar Albalawi *

College of Computing and Information Technology, University of Tabuk, Tabuk 71491, Saudi Arabia; 452009920@stu.ut.edu.sa (A.A.)

* Correspondence: 452009749@stu.ut.edu.sa (E.A.); ualbalwi@ut.edu.sa (U.A.)

Abstract

The rapid expansion of the Internet of Things (IoT) has led to billions of interconnected devices generating and exchanging sensitive data across diverse domains, which introduces challenges in identity management (IdM) regarding privacy, scalability, and verifiability. While blockchain technology provides decentralization and tamper resistance, its transparency and increasing on-chain storage demands make it unsuitable for large-scale IoT identity ecosystems. To overcome these challenges, IoT-SBIdM is proposed as a lightweight, privacy-preserving, and stateless blockchain-based identity management framework designed for IoT environments. This framework incorporates Elliptic Curve Cryptography (ECC)-based accumulators and Zero-Knowledge Proofs (ZKPs) to facilitate selective disclosure, enabling entities to prove credential authenticity without exposing sensitive identity information. Furthermore, the framework adopts W3C-compliant Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs) to promote interoperability and user-controlled identity ownership. The experimental results indicate that IoT-SBIdM achieves efficient smart contract execution by reducing gas costs through optimized registry logic. Moreover, the system maintains a compact block size of only 45 MB at higher block heights, outperforming comparable schemes in storage efficiency by achieving a 55% reduction relative to recent models and an approximate 94% reduction relative to older systems, thereby demonstrating superior scalability and storage efficiency, making it suitable for identity management solutions for IoT environments.

Keywords: blockchain-based identity; identity management (IDM); internet of things (IoT); selective disclosure; decentralized identifiers; verifiable credentials

MSC: 68N30; 68M14; 68M25



Academic Editor: Hui Cui

Received: 21 January 2026

Revised: 11 February 2026

Accepted: 13 February 2026

Published: 18 February 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

The Internet of Things (IoT) represents a transformative technological paradigm that interconnects billions of smart devices capable of collecting, processing, and exchanging data autonomously. These devices, embedded with sensors, processors, and communication interfaces, have revolutionized industries such as healthcare, manufacturing, smart cities, and logistics by enabling automation, predictive analytics, and intelligent decision-making [1]. Through such capabilities, IoT enhances energy optimization in smart grids, supports predictive maintenance in Industrial IoT (IIoT), and improves patient outcomes in connected healthcare environments [2]. However, the rapid expansion of IoT ecosystems

introduces significant security challenges, particularly in terms of authentication, identity management, privacy, scalability, and security [3–5]. As billions of devices interact within open and distributed networks, ensuring trustworthy digital identities becomes a cornerstone for secure communication and data integrity. Without robust identity management mechanisms, IoT systems are vulnerable to identity spoofing, unauthorized access, and data breaches [1,6].

Digital identities are essential for trusted transactions and thriving business opportunities [7]. In this context, Identity Management Systems (IdMSs) play a crucial role in establishing and maintaining secure, interoperable digital identities. According to the National Institute of Standards and Technology (NIST), digital identity represents “the unique representation of a subject engaged in an online transaction” [8]. IdMSs authenticate and authorize entities using identifiers (e.g., UserID, device IDs, usernames, email), credentials (e.g., certificates, tokens, biometrics), and attributes (e.g., roles, privileges) [6]. IdMS encompasses the management of identity information through a series of operations, including registration, updates, revocation, and identity lookup. For IoT ecosystems, these systems must support dynamic, scalable, and decentralized operations capable of managing vast numbers of devices and their identity relationships efficiently.

Traditional centralized IdMSs suffer from scalability and security limitations when applied to IoT contexts. Centralized identity providers act as single points of failure and are susceptible to cyberattacks, data breaches, and performance bottlenecks [3,4]. Moreover, centralized architectures often lead to privacy violations through the excessive collection and misuse of personally identifiable information (PII) [6,7].

As IoT continues to expand, facilitating billions of interconnected devices, the demand for robust identities becomes critical. An IoT device may also have a digital identity, representing it by another set of attributes, such as name, Media Access Control (MAC) address, Firmware, device type, category, vendor, and model, among others [1,9]. To address these limitations, the research community has increasingly turned toward decentralized identity frameworks, where identities are owned and controlled by their subjects rather than by centralized authorities [3,7,10,11]. Among these, blockchain technology, a distributed and tamper-resistant ledger, offers promising capabilities for secure and verifiable identity management in IoT networks [12,13]. Blockchain provides essential properties such as immutability, transparency, auditability, decentralization, tamper resistance, public verifiability, and trustlessness [10,14,15].

In decentralized identity management, blockchain enables IoT devices to register, verify, and manage identities autonomously, thereby reducing dependency on centralized servers and enhancing system resilience [4]. When combined with Zero-Knowledge Proofs (ZKPs), blockchain allows for privacy-preserving authentication, ensuring that devices can prove the validity of their credentials without disclosing sensitive data [16–18].

Furthermore, efforts by the World Wide Web Consortium (W3C) have standardized decentralized identity frameworks through the development of Decentralized Identifiers (DIDs), Verifiable Credentials (VCs), and Verifiable Presentations (VPs) [19,20]. DIDs are owner-controlled unique identifiers, while VCs are cryptographically signed attribute claims [21]. These standards enable interoperability and privacy-preserving verification, laying a strong foundation for scalable and trustworthy identity management in IoT [22].

Figure 1 illustrates the W3C Verifiable Credentials model, which involves four primary entities: the Issuer, who creates and issues VCs; the Holder, who stores them and presents VPs to the Verifier; the Verifier, who validates VPs; Verifiable Data Registry (VDR), which maintains verification records.

A robust Identity Management (IdM) architecture for IoT must satisfy several fundamental security and privacy requirements, including blindness, unforgeability, traceability,

unlinkability, revocability, and selective disclosure (SD), while ensuring compliance with W3C DID standards. Moreover, such a system must provide lightweight verification, anti-attack capabilities, and trustworthiness to accommodate the constraints of resource-limited IoT devices. Existing blockchain-based IdM frameworks, such as LSTIDM-SB [16], have attempted to address storage efficiency through RSA-based accumulators and fixed-length identity commitments. However, RSA-based cryptographic operations are computationally intensive and storage-heavy, rendering them impractical for low-power IoT devices.

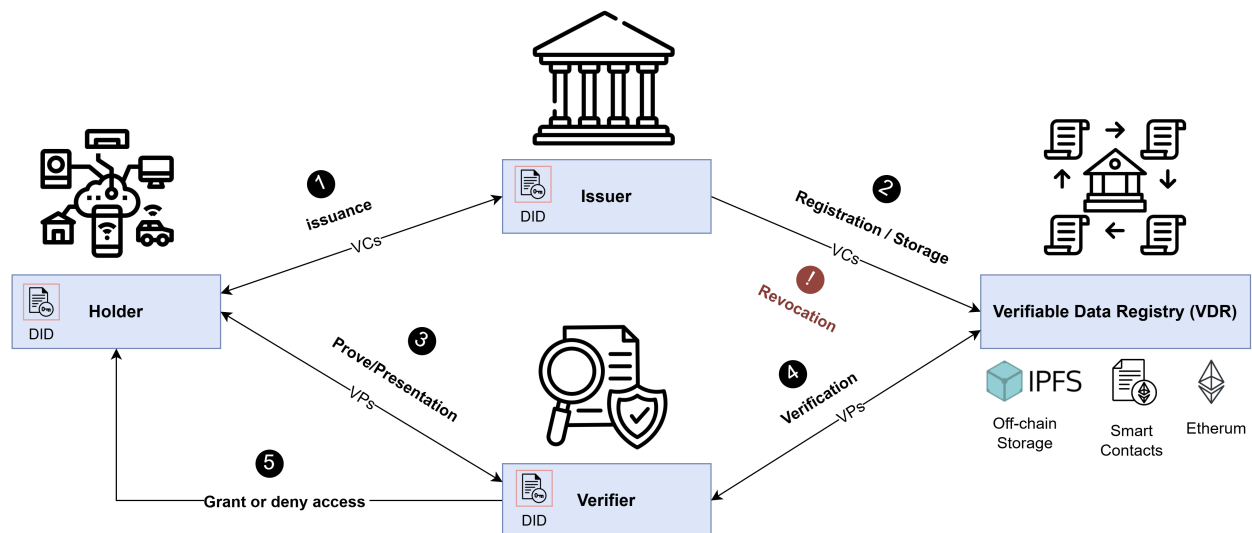


Figure 1. Generic Identity Management (IdM) Workflow Utilizing W3C Verifiable Credentials and Decentralized Identifier standards (VCs: Verifiable Credentials, VPs: Verifiable Presentations, DID: Decentralized Identifier, IPFS: InterPlanetary File System).

To overcome these limitations, this research proposes a stateless blockchain-based identity management system (IoT-SBIdM) specifically tailored for IoT ecosystems. The proposed framework integrates Elliptic Curve-Based Accumulators (ECA) to improve scalability and reduce computational cost, while employing a stateless blockchain with a Proof of Stake (PoS) consensus mechanism to ensure energy efficiency and high throughput. The motivation for this research stems from the urgent need to develop a lightweight, secure, scalable, and privacy-preserving W3C-compliant decentralized identity management system capable of supporting large-scale IoT networks by improving upon existing solutions while aiming for minimal computational and storage overhead.

The main contributions of this work to blockchain-based identity management for IoT can be summarized as follows.

- We propose a generic Identity Management (IdM) workflow based on the W3C Verifiable Credentials and Decentralized Identifier standards, which will serve as the foundation for our proposed system.
- End-to-end IoT IdM with W3C DID/VC/VP: A practical pipeline that issues and verifies Verifiable Credentials and Presentations anchored in W3C DIDs, supports selective disclosure (ecdsa-sd-2023), and integrates ZK membership proofs for privacy-preserving validation.
- EC-based membership accumulator with split verification: An elliptic-curve accumulator anchored on-chain, paired with Groth16 verification and an off-chain/on-chain split verification flow to minimize latency while retaining on-chain verifiability. This change improves computational efficiency and reduces the burden on blockchain storage.

- Privacy-Preserving Authentication via ZKP and SD: By incorporating ZKPs into the issuance and verification processes, the system enables secure and private identity validation, supporting selective disclosure and compliance with W3C standards.
- Secure communication channel via W3C DID's keyAgreement: A JSON Web Encryption (JWE) style used for encrypting the communication channel that leverages W3C DID's keyAgreement keys, using Elliptic Curve Diffie–Hellman Ephemeral-Static (ECDH-ES) for shared secret derivation and AES-256-GCM for confidentiality and integrity during end-to-end communication.

Building on the aforementioned contributions, IoT-SBIdM is, to the best of our knowledge, the first lightweight framework that unifies stateless blockchain-based identity management, W3C DIDs/VCs, ECC-based accumulators, ZKP-based selective disclosure, off-chain IPFS storage, and DID-based secure communication via “KeyAgreement” service in a single end-to-end IoT system. The framework supports the complete identity lifecycle and is validated through a working prototype on an Ethereum-compatible blockchain with a GUI and real IoT hardware (ESP32).

The rest of this research is organized as follows. Section 2 provides a review of the recent existing solutions, focusing on identity management approaches in IoT environments, including both centralized and decentralized models. Section 3 outlines the proposed framework's methodology, detailing the system's architecture across the application. Section 4 presents the experimental results and discussion. Finally, Section 5 concludes the research by summarizing key contributions and offering recommendations for future enhancements in decentralized identity systems tailored for IoT ecosystems.

2. Related Work

The integration of blockchain and Zero-Knowledge Proofs (ZKPs) within identity management systems (IDMs) for the Internet of Things (IoT) has received substantial attention due to its potential to enhance security, privacy, and scalability in decentralized environments. This section reviews prominent blockchain–ZKP-based identity management frameworks and identifies their limitations relative to the proposed IoT-SBIdM system.

To begin with, Yu et al. [15] proposed a blockchain-based anonymous authentication with selective revocation for Smart Industrial Applications (BASS), which addresses the significant challenge of privacy disclosure in smart industrial environments like smart vehicles. Their work introduces an efficient selective revocation mechanism, which relies on dynamic accumulators and the signature algorithm from Pointcheval and Sanders (PS). This methodology allows for attribute privacy, credential soundness, multishowing-unlinkability, and selective revocation, thereby solving the credential revocation issue that has been a longstanding problem in anonymous authentication protocols. However, BASS does not support W3C standards, such as DIDs and VCs, limiting interoperability and cross-platform scalability in IoT ecosystems.

Building upon the need for decentralized privacy-preserving solutions, Ra et al. [23] introduced the Verifiable Anonymous Identity Management (VAIM) system, which leverages blockchain and ZKP to provide a decentralized, user-centric approach to identity management. By using blockchain to create a transparent yet private ledger, VAIM addresses the challenges posed by centralized identity management systems. The model promotes self-sovereign privacy, accountability, and data access control through private keys and ZKP-based certificates. However, while VAIM strengthens anonymity, it lacks efficient verifiability and traceability mechanisms, constraining its applicability in multi-domain IoT environments requiring dynamic identity revocation and validation.

In a further advancement, Bao et al. [12] proposed PBidm, a privacy-preserving blockchain-based identity management system for IIoT, recognizing the trade-off between

excessive anonymity and centralization. PBidm balances anonymity and traceability while adhering to W3C DID and VC standards, enabling self-sovereign identity ownership. Despite this, PBidm’s verification process remains computationally intensive, and its reliance on on-chain operations introduces latency and scalability bottlenecks for large-scale IoT networks.

To address scalability and flexibility challenges, FLEXDID, presented by Bian et al. [3] focuses on providing a flexible and scalable decentralized identity management system for IIoT environments. FLEXDID supports both vertical and horizontal credential disclosure while providing efficient credential revocation and optimizations such as batch verification, making it an adaptable solution for IIoT. Nonetheless, FLEXDID is not blockchain-dependent, thereby lacking on-chain verifiability, immutability, and trust anchoring, which are essential for ensuring data integrity and auditability in distributed IoT systems.

Finally, Zhang et al. [16] proposed Stateless Blockchain-Based Lightweight Identity Management (LSTIDM-SB), which addresses storage overhead and security vulnerabilities in IIoT by introducing RSA-based accumulators and length-constant identity commitments (IDC). The system also introduces a novel consensus algorithm (SPoW) and a non-interactive proof protocol (NI-PoE) to improve block creation and validation efficiency. While this approach enhances anti-attack capability and lightweight operation, it does not incorporate selective disclosure mechanisms, which are critical for maintaining user privacy and enabling fine-grained access control in IoT ecosystems.

Table 1 presents a comparative analysis between the proposed IoT-SBIdM system and recent state-of-the-art IDM frameworks. IoT-SBIdM surpasses prior works by integrating W3C standards (DID, VC, VP), supporting selective disclosure, credential validity periods, and secure key agreement, while maintaining traceability, revocability, and lightweight operation through its elliptic-curve accumulator and split verification design.

Table 1. Comparison of IoT-SBIdM with Recent Related Work.

Feature	IoT-SBIdM (Ours)	T.Ramírez-Gordillo et al. (2025) [5]	LSTIDM-SB (2024) [16]	PBIdm (2023) [12]	TAB-SAPP (2023) [24]	CanDID (2021) [11]	BASS (2020) [15]	Coconut (2020) [25]
Blindness	✓	✓	✓	✓	✓	✓	✓	✓
Unlinkability	✓	✓	✓	✓	✓	✓	✓	✓
Unforgeability	✓	✓	✓	✓	✓	✓	✓	✓
Traceability	✓	✓	✓	✓	✗	✗	✗	✗
Revocability	✓	✓	✓	✓	✗	✗	✗	✗
Efficient Verifiability	✓	✓	✓	✗	✗	✗	✗	✗
W3C DID standard	✓	✓	✓	✓	✗	✓	✗	✗
W3C VC standard	✓	✓	✓	✓	✗	✗	✗	✗
DID’s Key Agreement	✓	✗	✗	✗	✗	✗	✗	✗
DID’s Service	✓	✓	✗	✗	✗	✗	✗	✗
Selective Disclosure	✓	✗	✗	✗	✗	✗	✗	✓
Cred. Validity Period	✓	✗	✗	✗	✗	✗	✗	✗
Trustworthiness	✓	✓	✓	✓	✗	✗	✗	✗
Anti-attack Capability	✓	✓	✓	✗	✗	✓	✗	✗
Lightweight	✓	✗	✓	✗	✓	✗	✗	✗

The symbol ✓ indicates compliance with the listed feature; ✗ indicates non-compliance.

3. Proposed System

This chapter outlines the methodology for developing and implementing the proposed stateless blockchain-based identity management system for IoT networks (IoT-SBIdM). The approach encompasses system design, implementation of key components, and integra-

tion of various technologies to achieve a secure, scalable, and privacy-preserving identity management framework. The methodology is structured to provide a clear roadmap for constructing the system, detailing the procedures, tools, and technologies involved. To demonstrate the real-world integration and effectiveness of our system in validating VPs prior to granting access or permissions, we employed an ESP32 PLUS development board (smart farm kit) as the test platform.

3.1. Proposed System Architecture

The IoT-SBIdM system is composed of three integrated layers: (1) Application Layer, (2) Issuance and Verification Layer, and (3) Distributed Ledger Layer, as illustrated in Figure 2. Before the identity management process can begin, all participating entities, including the Holder, Issuer, and Verifier, must first obtain a DID to establish their cryptographic identity in the ecosystem.

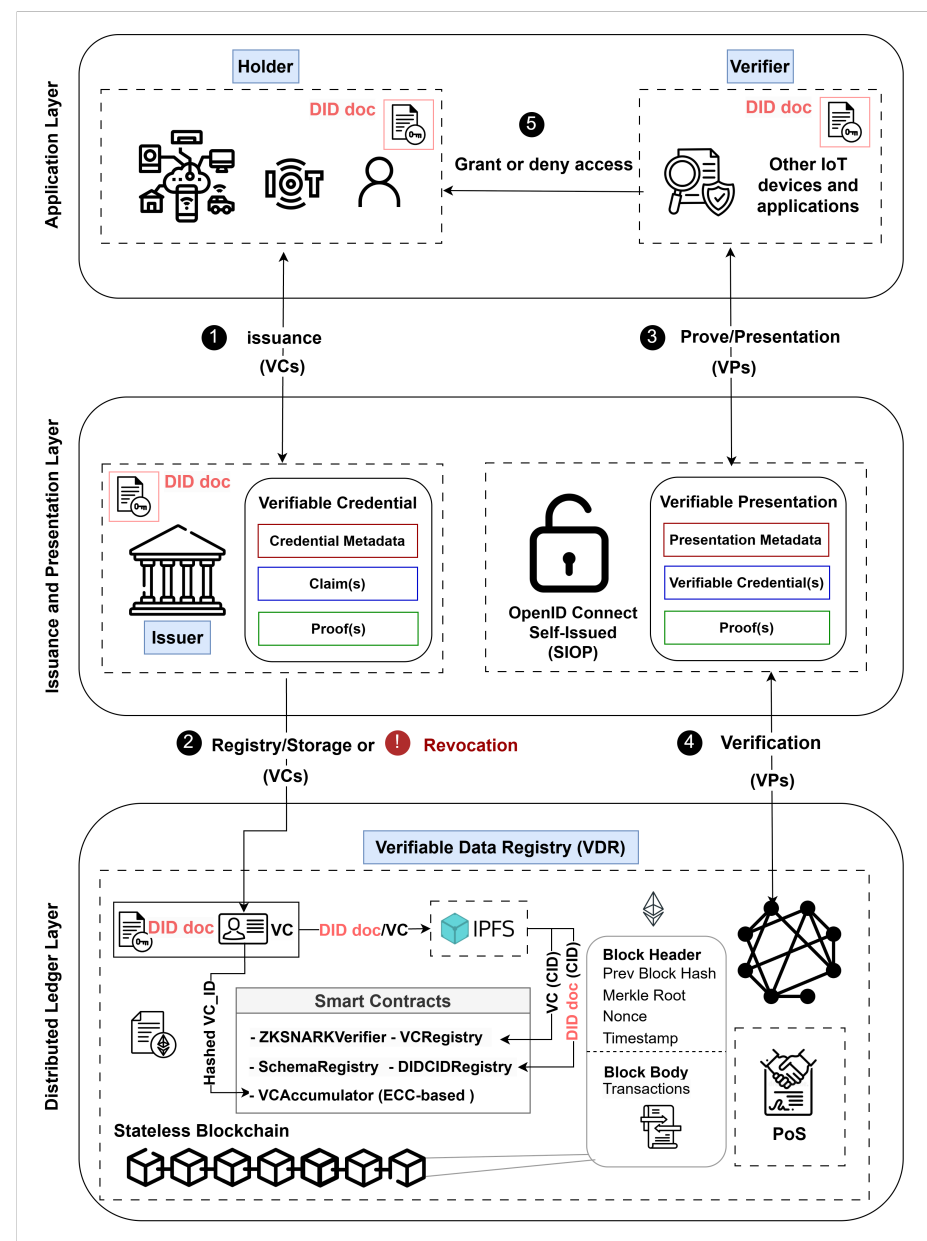


Figure 2. The proposed system architecture of IoT-SBIdM framework, including three layers: application layer, issuance and verification layer, and distributed ledger layer, and six phases: obtain DID, issue VC, register VC to VDR, verify VP, prove VP, accept VP and grant access.

When connecting to the IoT-SBIdM system for the first time, each entity (Holder, Issuer, Verifier) generates a W3C-compliant DID document, each of which includes several key elements:

- Public Key(s): used for signature verification and authentication.
- Authentication Method: defines how the entity can prove control of the DID.
- Assertion Method: allows entities to issue verifiable statements or credentials.
- Key Agreement: enables secure communication through shared secret derivation (e.g., ECDH-ES).
- Service Endpoints: describe communication or verification endpoints for the entity.

Once generated, the DID document is stored off-chain in the InterPlanetary File System (IPFS) to ensure decentralization and reduce on-chain data overhead. The resulting Content Identifier (CID), a unique hash of the stored DID document, is then committed on-chain to a smart contract, forming an immutable reference for DID verification. This registration process ensures that all entities have a verifiable on-chain identity anchored in cryptographic trust, serving as the foundation for subsequent credential issuance and verification.

3.1.1. Application Layer

At the top level, the Application Layer represents the interaction point between real-world entities such as IoT devices, smart homes, connected vehicles, industrial equipment, and end-users, collectively referred to as Holders. Once DIDs are established, Holders request Verifiable Credentials (VCs) from trusted Issuers (Phase 1), which can later be used to access IoT applications or services. During access requests, Holders generate and present Verifiable Presentations (VPs) (Phase 3) to Verifiers, who evaluate their validity using the blockchain-based Verifiable Data Registry (VDR). Upon successful verification, the Verifier grants or denies access (Phase 5) based on the credential's authenticity and revocation status. This process provides secure, decentralized, and privacy-preserving authentication between IoT entities.

3.1.2. Issuance and Verification Layer

This intermediate layer manages the generation, validation, and presentation of verifiable credentials using decentralized identifiers and Zero-Knowledge Proofs. This layer thus bridges the application entities and the underlying blockchain infrastructure, ensuring that all credential interactions remain verifiable.

1. Credential Issuance (Phase 1):

When a Holder requests identity credentials, the Issuer constructs a VC that includes:

- (a) Credential Metadata (e.g., issuance timestamp, issuer ID reference),
- (b) Claims (asserted identity attributes of the Holder),
- (c) Proofs (cryptographic signatures ensuring authenticity and integrity).

The Issuer transmits the VC to the Holder while simultaneously recording a reference in the VDR for revocation tracking and verifiability (Phase 2).

2. Credential Presentation and Verification (Phases 3–4):

When authentication is required, the Holder can use OpenID Connect Self-Issued OpenID Provider (SIOP) to construct a VP. The VP includes selective disclosure parameters, allowing the Holder to reveal only essential credential data to the Verifier. The Verifier then checks the VP against the VDR to confirm its validity, authenticity, and non-revocation before granting access.

3.1.3. Distributed Ledger Layer

At the foundation, the Distributed Ledger Layer anchors trust, integrity, auditability, and decentralized verification of all credentials using blockchain technology. It minimizes on-chain data storage while maintaining full verifiability of all credential operations. Core functionalities include:

- **Off-chain Storage via IPFS (Phase 2):** VCs and DID documents are stored in IPFS, decoupling data from the blockchain to preserve decentralization. The resulting CIDs serve as unique references to the off-chain data.
- **Hashed Credential Commitment:** The Issuer commits the hashed VC identifier and related metadata to the blockchain through ECC-based accumulators. This cryptographic accumulator allows lightweight and efficient verification of credential inclusion or revocation without requiring full data exposure.
- **Consensus Mechanism:** The blockchain operates under a PoS consensus algorithm, ensuring energy efficiency, rapid transaction finality, and resilience against centralization risks.

When a Verifier initiates a credential validation request (Phase 4), blockchain nodes independently verify the authenticity of the credential using the stored hash and accumulator states, thereby eliminating the need to retain full identity datasets or reprocess complete blockchain history.

3.2. Design Goals

We outline the design goals of our identity system.

1. **Blindness:** This feature ensures that the Issuer cannot correlate the VC it issues with a specific Holder, preserving the Holder's privacy during issuance.
2. **Unforgeability:** Guarantees that only authorized Issuers can create valid VCs, which can be cryptographically verified by Verifiers. Data integrity proofs ensure that credentials issued by credential services are valid and prevent unauthorized issuance through anchoring in the accumulator contract.
3. **Traceability:** Traceability allows the Issuer to later associate a VC with the original Holder, if necessary for auditing, regulatory compliance, or dispute resolution. DID registration and device inventory logs link credentials to on-chain ownership and audit metadata.
4. **Unlinkability:** Ensures that multiple VPs created by the same Holder cannot be linked together by different Verifiers.
5. **Revocability:** Ensures that VCs can be revoked by the Issuer if compromised, expired, or no longer valid.
6. **Trustworthiness:** The system is built on open standards and cryptographic integrity, ensuring reliability and transparency. Through the implementation of KeyAgreement and Service properties in the W3C DID standard, these elements reinforce this goal [20]. The DID acts as a single source of truth for both secure communication keys and authoritative service locations, all protected by the same cryptographic integrity guarantees that underpin the rest of the system.
 - (a) **"KeyAgreement" keys:** Prove that the DID controller can establish authenticated, encrypted channels. When devices or services pull the DID document, they learn exactly which public key to use for Diffie-Hellman or ECDH exchanges. Because the key material is signed and anchored on-chain through the DIDCIDRegistry and IPFS workflow, any tampering is detectable. Provisioning messages, VC requests, and actuator commands can be wrapped in

authenticated encryption tied to the DID, improving integrity, confidentiality, and non-repudiation.

- (b) Service endpoints (“service” property): Bind specific network addresses (type, serviceEndpoint, optional routing information) to the DID under the same cryptographic proof. Holders or Verifiers can discover official Issuer APIs (HTTPS), MQTT brokers, or revocation feeds directly from the DID document, ensuring they communicate only with the correct DID controller. Updates require signed on-chain transactions, yielding tamper-evident service discovery without relying on external DNS or manual configuration trust assumptions.
- 7. Anti-Attack Capability: The system leverages blockchain-backed VDRs, cryptographic proofs, and secure VPs to resist attacks.
- 8. Lightweight: The identity system is optimized for low-resource devices by minimizing processing, memory, and power requirements. It employs efficient cryptographic primitives such as ECC, stateless blockchains, and minimal on-chain storage through decentralized file systems like IPFS for off-chain data.
- 9. Efficient Verifiability: The process of verifying a VP must be fast and scalable to support real-time IoT operations. Groth16 proofs produced by the membership circuit are checked off-chain and on-chain, and accumulator membership queries remain constant time.
- 10. Selective Disclosure: Holders can generate VPs that reveal only specific claims from a VC, hiding unnecessary or sensitive information. Selective-disclosure suites are integrated into the VP generation services and Holder UI, enabling fine-grained claim minimization.

3.3. Data Structure

Building on the DID and VC foundations introduced in Section 3.1, this section details the data structures used throughout the IoT-SBIdM identity lifecycle. In the proposed system, each participating entity, such as Holder, Issuer, and Verifier, is assigned a decentralized identifier consisting of three logical components: the DID scheme (did:), the Ethereum-based method (ethr) extended with the IoT-SBIdM namespace, and an Ethereum account-based method-specific identifier. This DID anchors the entity’s cryptographic identity and corresponds to a DID Document previously generated and stored through the on-chain/off-chain mechanism described in Section 3.1.

Each DID Document contains the public verification keys, authentication and assertion interfaces, and service endpoints defined by the W3C DID Core standard. IoT-SBIdM maintains two “Multikey” entries per entity, one dedicated to verification operations (e.g., VC/VP signatures) and another for key-agreement operations such as ECDH, each encoded using “publicKeyMultibase” for cross-suite compatibility.

The VC extends this identity layer by embedding authoritative statements about an IoT device, including its model, serial number, and operational capabilities. These credentials incorporate issuer-controlled metadata and are signed using the “ecdsa-sd-2023” cryptosuite to enable selective disclosure. The credential schema is stored off-chain on IPFS and referenced using a CID within the VC’s “@context”, ensuring decentralized and verifiable schema integrity.

For privacy-preserving authentication, IoT-SBIdM employs VPs, through which the holder transmits either the complete VC or a selectively disclosed subset of its claims. The VP is signed by the holder using the “ecdsa-2019” cryptosuite and bound to a verifier-issued challenge to prevent replay. To further enhance privacy, the VP includes a Groth16 zero-knowledge proof demonstrating the validity of hidden VC fields, such as accumulator membership, without exposing the underlying data.

Together, the DID Document, VC, and VP structures fully comply with W3C standards and form the core data model that enables decentralized, privacy-preserving device identity within IoT-SBIdM. The integrated data structure is illustrated in Figure 3.

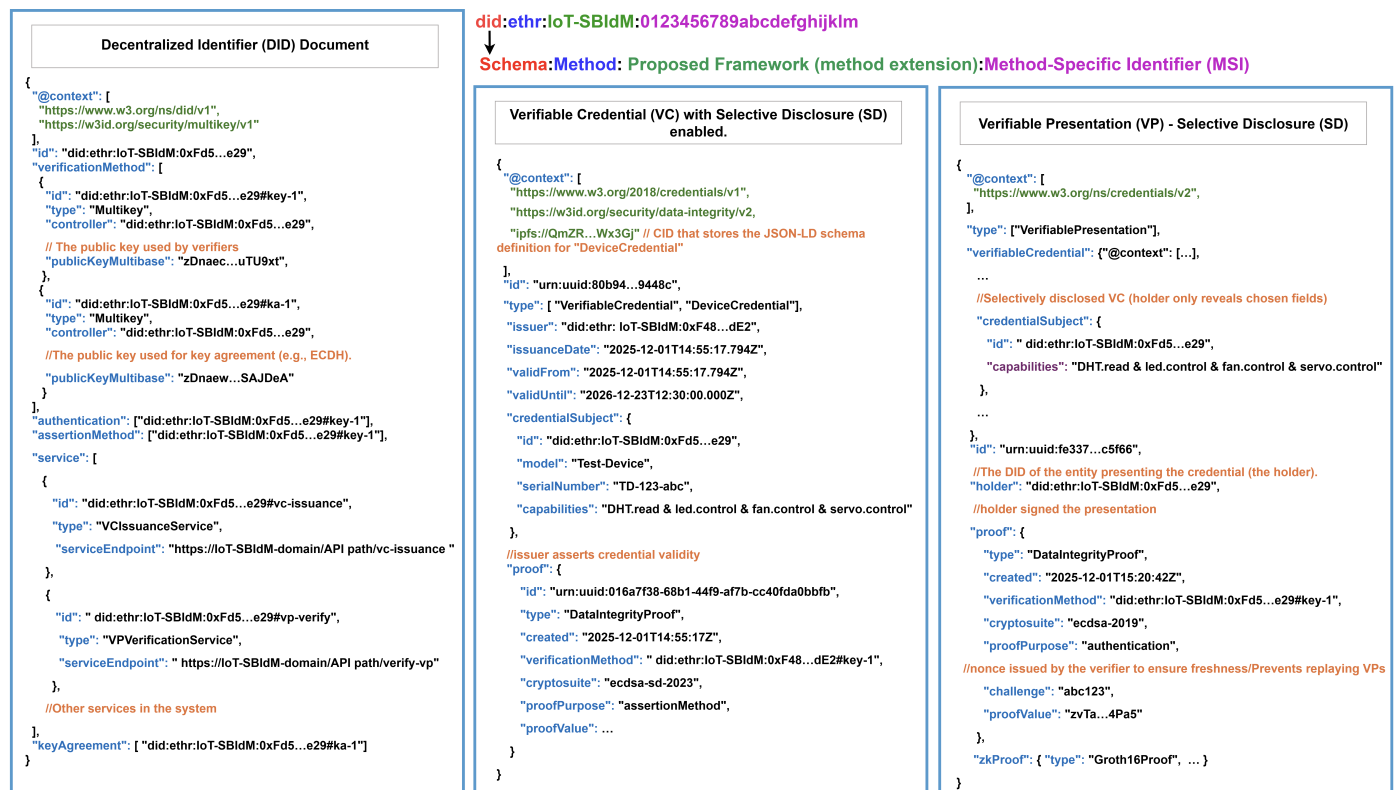


Figure 3. Data structure of the IoT-SBIdM system, including the DID Document, VC with selective disclosure support, and VP containing disclosed claims and associated zero-knowledge (Groth16) proofs. The identifier at the top illustrates the DID scheme (did), method (ethr), proposed method extension (IoT-SBIdM), and method-specific identifier.

3.4. Environment Specification and Implementation Setup

The implementation of the proposed IoT-SBIdM system leverages a combination of blockchain, cryptographic, and IoT technologies to realize a privacy-preserving, decentralized, and verifiable identity framework. The following tools, frameworks, and hardware components were employed to develop and validate the system in a local environment.

- **Blockchain Platform:** The system is deployed on the Ethereum blockchain, selected for its mature ecosystem and the transition to the PoS consensus protocol, which enhances scalability and sustainability while maintaining network security. Smart contracts written in Solidity (v0.8.21) manage DIDs, VCs, and ZKP verification to ensure decentralized trust and non-repudiation.
- **Cryptographic Libraries, suites and algorithms:** Several cryptographic libraries, suites, and algorithms were utilized, including snarkjs (<https://github.com/iden3/snarkjs>) and Circomlib (<https://github.com/iden3/circomlib>), cryptographic circuit templates for ZKP generation and verification [26], ECDSA-2019-cryptosuite (<https://github.com/digitalbazaar/ecdsa-2019-cryptosuite>) and ECDSA-sd-2023-cryptosuite (<https://github.com/digitalbazaar/ecdsa-sd-2023-cryptosuite>) for elliptic curve and selective disclosure mechanisms. In addition, the data-integrity (<https://github.com/digitalbazaar/data-integrity>) library offers a ready-made DataIntegrityProof suite that integrates with jsonld-signatures for minting and verifying W3C VCs using modern cryptosuites. Likewise, ecdsa-multikey (<https://github.com/>

[digitalbazaar/ecdsa-multikey](#)) provides standardized NIST P256 ECDSA key generation and management in the W3C Multikey format, enabling direct insertion into DID Documents and VC proofs without custom serialization.

These libraries and suites are integrated from open-source repositories to facilitate secure proof generation and selective claim sharing [27]. AES-256-GCM algorithms with JSON Web Encryption (JWE) standard were used for secure data transmission and confidentiality.

- InterPlanetary File System (IPFS): Serves as a decentralized off-chain storage solution for DID Documents and Verifiable Credentials. This ensures tamper-proof, scalable, and content-addressable storage while preventing blockchain data overload [28]. Only the cryptographic hash (CID) references are maintained on-chain for verifiability.
- OpenID Connect (Self-Issued OpenID Provider-SIOP): The system employs OpenID Connect (SIOP) for privacy-preserving VPs, enabling Holders to generate and present privacy-preserving VPs compliant with SSI principles and supporting ZKP-based claims. This allows holders to prove claims without disclosing unnecessary personal information [29,30].
- Programming Languages: The prototype is implemented primarily using JavaScript as the core language for frontend and backend logic, Solidity for on-chain smart contract development, and Circom as a domain-specific language employed for ZKP circuit generation and verification workflows. Arduino was used to program the ESP32 PLUS development board.
- Runtime Environment: Node.js (v24.11.1), a JavaScript runtime, was utilized for back-end execution and Representational State Transfer (REST) Application Programming Interface (API) services.
- Development Frameworks: Truffle (v5.11.5) and Ganache (v2.7.1) (<https://archive.trufflesuite.com/>) are used for local Ethereum network emulation, contract testing, and debugging. Express.js (v4.18.2), a web application framework built on Node.js, provides routing, middleware, and HTTP utilities.
- Development Libraries: Web3.js (v4.16.0) and Ethers.js (v6.14.1) provide the bridge between the blockchain and client-side components, enabling secure interaction with deployed smart contracts. React.js (v19.1.0), a frontend library, is used for building user interfaces.
- Wallet Integration: Leverages MetaMask, a browser-based Ethereum wallet extension, for authentication, signing, and secure transaction handling.
- Hardware and Software: Standard personal computers were used for development and simulation (i.e., Intel(R) Core(TM) Ultra 7 155H 3.80 GHz, 16.0 GB RAM, running Windows 11 OS).
- IoT Demonstration Node (Keystudio ESP32 Smart Farm Kit (<https://docs.keystudio.com/projects/KS0567/en/latest/wiki/>)): A real-world device called the ESP32 Smart Farm Kit was utilized in our experiments. It shipped with the ESP32 PLUS development board (integrated with the ESP32-WROOM-32 module) with a 240 MHz WiFi/BT CPU and 4 MB of flash memory. The ESP32's WiFi operates based on IEEE 802.11 standards, [31] utilizing the 2.4 GHz frequency band for wireless communication. The ESP32 Smart IoT Farm firmware was implemented using the Arduino IDE to validate the IoT-SBIDM system's integration with physical IoT devices. The ESP32 acts as both a Holder and a Verifier within the identity management workflow. This device performs several key identity and operational functions:
 - Create and store public and private key pairs locally in Non-Volatile Storage (NVS), a flash-based key-value store on the ESP32.
 - Obtain a DID by sending a request to the system.

- Obtain a VC issued by the system's issuer.
- Generate and present a VP for authentication and data exchange.
- Send sensor readings (e.g., from DHT temperature and humidity sensors) securely to the IoT-SBIdM system using Message Queuing Telemetry Transport (MQTT) mosquitto (https://hub.docker.com/_/eclipse-mosquitto) broker.
- Await and process control commands (e.g., activating actuators such as LED lights, fans, and servo motors) issued through verified identities.

3.5. System Modeling

To illustrate the logical structure and interactions among the core components of the IoT-SBIdM system, Unified Modeling Language (UML) diagrams were employed [32]. These diagrams provide a visual specification of the dynamic workflows between entities and the internal architecture of the smart contract layer, ensuring clarity in protocol design and implementation logic.

Figure 4 demonstrates how the ESP32 PLUS board initiated a VC issuance request to IoT-SBIdM's backend service that is handled by the Issuer. The process begins with the device loading its DID document and cryptographic keys from local storage to ensure identity readiness. The device then submits an issuance request containing its DID and encrypted claims to the Issuer endpoint, which forwards the request to the credential issuance engine for validation. The engine verifies the device's DID anchoring via the on-chain DID Registry smart contract and, upon successful confirmation, constructs the verifiable credential. A hash of the credential identifier is then registered in the on-chain membership accumulator to support revocation and unlinkability, while the signed credential payload is stored off-chain using IPFS. The resulting content identifier (CID) is recorded on-chain in the VC Registry to bind the credential reference to the holder. Finally, the issuer returns the issued VC and CID to the ESP32 device, which securely caches the credential locally for subsequent presentation and verification. Any failure in DID verification, accumulator registration, or off-chain storage aborts the issuance process and requires corrective action before the process can be retried.

Figure 5 illustrates the complete end-to-end interaction lifecycle between the ESP32 device and the IoT-SBIdM architecture, detailing the workflow from initial bootstrapping to steady-state operation. Upon power-on, the device performs diagnostic routines and engages the lifecycle manager to initialize peripherals and verify readiness. Once stabilized, the ESP32 secures its identity by loading cryptographic keys and DID context via the identity and credential service provided by the device file manager, establishing a trusted foundation before any external communication. With identity assurance in place, the device connects to the IoT-SBIdM backend to request verifiable credential issuance. The backend coordinates credential anchoring across on-chain smart contracts and distributed off-chain storage via IPFS, ensuring durable state persistence, revocation support, and cross-domain verifiability. Upon confirmation, the issued VC is securely cached locally on the ESP32, activating its operational profile. The lifecycle manager then transitions the device into steady-state operation, where telemetry exchange, control interactions, and verifiable presentation flows leverage the locally stored VC as a cryptographic trust anchor. This stored credential supports routine authentication and presentation workflows, eliminating repeated provisioning overhead and enabling continuous operation until revocation, policy changes, or a factory reset necessitate re-execution of the provisioning process.

Similarly, Figure 6 presents the system's Solidity smart contract class diagram showing classes such as VCRegistry, DIDCIDRegistry, SchemaRegistry, VCAccumulator, and ZKSNARKVerifier. The composition relationship (strong ownership) between VCAccumulator and Point struct implies that VCAccumulator contains or owns Point,

and if VCAccumulator is destroyed, the Point objects go with it. An aggregation relationship (weak ownership) between the SchemaRegistry and SchemaHashes struct means the SchemaRegistry contains or keeps references to many SchemaHashes, but those hashes could exist independently. If the registry is deleted, the hashes still exist elsewhere (in the IPFS node).

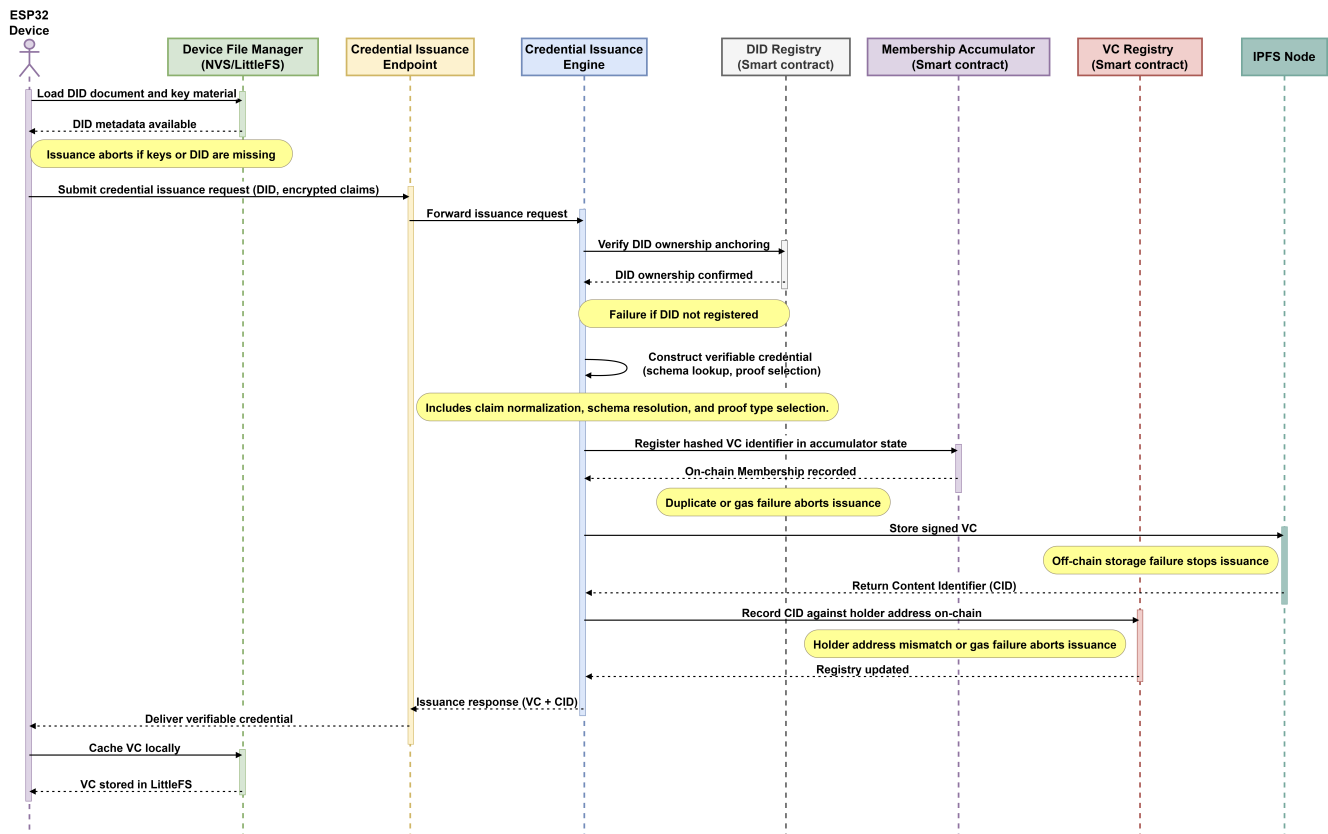


Figure 4. Sequence diagram illustrating the initiation of credential issuance request by ESP32 PLUS board.

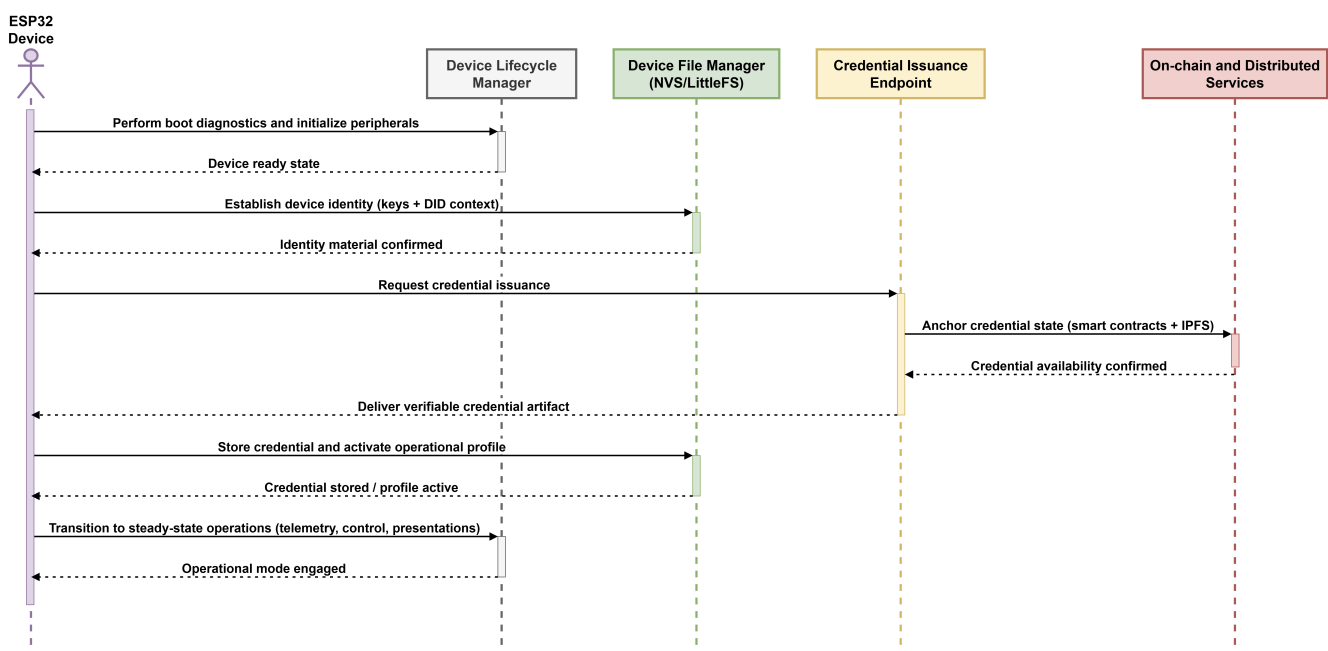


Figure 5. Sequence diagram summarizing the whole workflow of the IoT device (ESP32 Smart farm kit) with the IoT-SBIdM system.

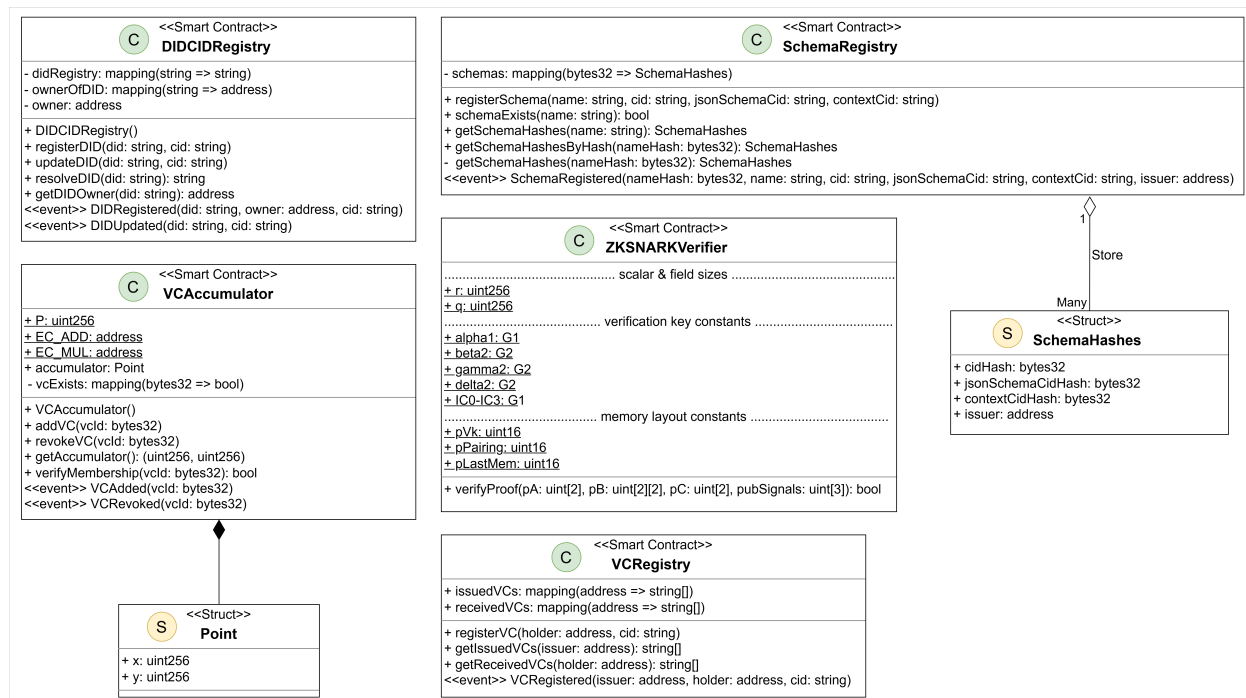


Figure 6. Class diagram of the IoT-SBIdM smart contract suite.

3.6. IoT-SBIdM Protocol

The operational logic of IoT-SBIdM is governed by a set of cryptographic and identity management protocols that define the interactions between entities. These include the processes for decentralized identifier generation, credential issuance, selective disclosure proof creation, verification, and revocation.

First, a device enrolls into the system through the DID Registration protocol (Algorithm 1), which constructs a DID document, stores it off-chain, and anchors a DID→CID binding on the blockchain. Once registered, the holder can request credentials using the Holder-side credential Issuance flow (Algorithm 2), where claims are packaged, encrypted via key agreement, and transmitted to the issuing service. The issuer processes this request through the Issuer-side credential Issuance protocol (Algorithm 3), which authenticates the request, constructs and signs a verifiable credential, anchors its identifier in the cryptographic accumulator, records metadata on-chain, and returns an encrypted response to the holder.

Algorithm 1 DID Registration Algorithm

Require: Subject public keys PK_{auth} , $PK_{agreement}$, method parameters $params$

Ensure: Registered DID identifier did

- 1: **procedure** REGISTERDID(PK_{auth} , $PK_{agreement}$, $params$)
- 2: $did = \text{BUILDDID}(params)$
- 3: $doc = \text{BUILDDIDDOCUMENT}(did, PK_{auth}, PK_{agreement})$
- 4: $cid = \text{STORETOIPFS}(doc)$
- 5: $tx = \text{REGISTERDIDONCHAIN}(did, cid)$
- 6: $\text{AWAITFINALITY}(tx)$
- 7: **return** did
- 8: **end procedure**

During authentication, devices construct privacy-preserving verifiable presentations. IoT-SBIdM supports two presentation modes depending on privacy requirements. The Full Disclosure Presentation workflow (Algorithm 4) generates a presentation containing the entire credential along with a holder-binding signature over the credential and verifier

challenge. For privacy-preserving authentication, the Selective Disclosure Presentation workflow (Algorithm 5) allows the holder to disclose only a minimal subset of attributes while proving, via a zero-knowledge proof, that these attributes originate from a valid, unmodified credential. These presentations are later validated by the verifier using the verification pipeline defined in Algorithm 6.

Algorithm 2 Credential Issuance (Holder-side)

Require: Holder DID did_h , holder secret sKA_h , issuer DID did_{issuer} , claims $claims$, auth token tok

Ensure: Signed verifiable credential VC

```

1: procedure ISSUEVCHOLDER( $did_h, sKA_h, did_{issuer}, claims, tok$ )
2:    $req = \text{BUILDISSUANCEREQUEST}(did_h, claims)$ 
3:    $KA_{issuer} = \text{RESOLVEKEYAGREEMENT}(did_{issuer})$ 
4:    $request_{enc} = \text{ENCRYPT}(req, KA_{issuer})$ 
5:    $issService_{API} = (did_h.service.type == IssuanceServ)$ 
6:    $\text{SENDREQUEST}(issService_{API}, request_{enc}, tok)$ 
7:    $res = \text{RECEIVERESPONSE}()$ 
8:    $VC = \text{DECRYPTANDPARSE}(res, sKA_h)$ 
9:    $\text{VERIFYISSUERSIGNATURE}(VC)$ 
10:   $\text{STORECREDENTIAL}(VC)$ 
11:  return  $VC$ 
12: end procedure
  
```

Algorithm 3 Credential Issuance (Issuer-side)

Require: Encrypted issuance request req , issuer signing key sk_{iss} , issuer secret sKA_{iss} , schema registry Sch

Ensure: Encrypted response containing signed VC

```

1: procedure ISSUEVCISSUER( $req, sk_{iss}, sKA_{iss}, Sch$ )
2:    $request = \text{DECRYPTANDPARSE}(req, sKA_{iss})$ 
3:   if  $\neg \text{VERIFYREQUEST}(request)$  then
4:     return ERROR
5:   end if
6:    $(claims, did_{holder}) = \text{EXTRACTPAYLOAD}(request)$ 
7:    $meta = (\text{ISSUANCETIME}(), \text{EXPIREDATE}(), \text{ID}())$ 
8:    $VC = \text{BUILDCREDENTIAL}(did_{holder}, claims, meta)$ 
9:    $VC_{signed} = \text{SIGNCREDENTIAL}(VC, sk_{iss})$ 
10:   $tx_1 = \text{ANCHORINACCUMULATOR}(VC_{signed}.id)$ 
11:   $\text{AWAITFINALITY}(tx_1)$ 
12:   $cid = \text{STORETOIPFS}(VC_{signed})$ 
13:   $tx_2 = \text{RECORDONCHAIN}(did_{holder}, cid)$ 
14:   $\text{AWAITFINALITY}(tx_2)$ 
15:   $response = \text{ASSEMBLERESPONSE}(VC_{signed})$ 
16:   $KA_{holder} = \text{RESOLVEKEYAGREEMENT}(did_{holder})$ 
17:   $response_{enc} = \text{ENCRYPT}(response, KA_{holder})$ 
18:  return  $response_{enc}$ 
19: end procedure
  
```

The Verifiable Presentation Verification procedure (Algorithm 6) validates structural correctness, temporal validity, digital signatures, off-chain Groth16 proofs, and on-chain accumulator membership. When credentials must be invalidated, the Revocation Protocol (Algorithm 7) authorizes the issuer, checks accumulator membership, updates the revocation state, and logs the event.

Algorithm 4 Verifiable Presentation Creation (Full)**Require:** Verifiable Credential VC , holder signing key sk_h , challenge nonce n **Ensure:** Full verifiable presentation VP_{full}

```

1: procedure CREATEVPFULL( $VC, sk_h, n$ )
2:    $proof = \text{SIGN}(VC \parallel n, sk_h)$ 
3:    $VP_{full} = \text{BUILDPRESENTATION}(VC, proof, n)$ 
4:    $\text{SENDPRESENTATION}(VP_{full})$ 
5:   return  $VP_{full}$ 
6: end procedure

```

Algorithm 5 Verifiable Presentation Creation (Selective Disclosure)**Require:** Verifiable Credential VC , disclosed fields F_{sd} , holder signing key sk_h , challenge nonce n **Ensure:** Selective-disclosure presentation VP_{sd}

```

1: procedure CREATEVPSELECTIVE( $VC, F_{sd}, sk_h, n$ )
2:    $claims_{sd} = \text{EXTRACTFIELDS}(VC, F_{sd})$ 
3:    $proof = \text{ZKPROVE}(claims_{sd}, VC, n, sk_h)$ 
4:    $VP_{sd} = \text{BUILDPRESENTATION}(claims_{sd}, proof, n)$ 
5:    $\text{SENDPRESENTATION}(VP_{sd})$ 
6:   return  $VP_{sd}$ 
7: end procedure

```

Algorithm 6 Verifiable Presentation Verification**Require:** Presentation VP , challenge nonce n , Groth16 verification key $VK_{groth16}$ **Ensure:** Boolean validity result

```

1: procedure VERIFYVP( $VP, n, VK_{groth16}$ )
2:   if  $VP.challenge \neq n$  then
3:     return INVALID
4:   end if
5:    $VCS = \text{EXTRACTCREDENTIALS}(VP)$ 
6:   if  $VCS = \emptyset$  then
7:     return INVALID
8:   end if
9:    $(PK_{holder}, PK_{issuer}) = \text{RESOLVEKEYS}(VP, VCS)$ 
10:  for all  $vc \in VCS$  do
11:     $v = \text{CHECKTEMPORALVALIDITY}(vc)$ 
12:    if  $\neg v$  then
13:      return INVALID
14:    end if
15:  end for
16:   $sigOk = \text{VERIFYSIGNS}(VP, VCS, PK_{holder}, PK_{issuer})$ 
17:  if  $\neg sigOk$  then
18:    return INVALID
19:  end if
20:   $zkOff = \text{GROTH16VERIFY}(VP.zkProof, VK_{groth16})$ 
21:  if  $\neg zkOff$  then
22:    return INVALID
23:  end if
24:   $h_{vcId} = \text{HASHIDENTIFIER}(VC.id)$ 
25:   $memberOk = \text{CHECKACCUMULATORMEMBERSHIP}(h_{vcId})$ 
26:   $zkOn = \text{VERIFYSNARKONCHAIN}(VP)$ 
27:  if  $memberOk \wedge zkOn \wedge sigOk \wedge zkOff$  then
28:    return VALID
29:  end if
30:  return INVALID
31: end procedure

```

Together, these algorithms define the end-to-end lifecycle of identities and credentials in IoT-SBIdM, ensuring integrity, privacy, selective disclosure, and blockchain-anchored accountability throughout the system.

Algorithm 7 Verifiable Credential Revocation

Require: Credential identifier $vcId$, issuer signing key sk_{iss}

Ensure: Revocation status *revoked*

```

1: procedure REVOKEVC( $vcId, sk_{iss}$ )
2:    $authOK = \text{VERIFYISSUERAUTH}(sk_{iss}, vcId)$ 
3:   if  $authOK$  then
4:      $h_{vcId} = \text{HASHIDENTIFIER}(vcId)$ 
5:      $isVCExisted = \text{VERIFYMEMBERSHIP}(h_{vcId})$ 
6:     if  $\neg isVCExisted$  then
7:       return ERROR("VC not found in accumulator")
8:     end if
9:      $isRevoked = \text{REVOKEVC}(h_{vcId})$ 
10:     $\text{APPENDREVOCATIONAUDITLOG}(vcId)$ 
11:    return true
12:  end if
13:  return false
14: end procedure
  
```

3.7. Threat Model

The threat model for IoT-SBIdM assumes that the issuer and the blockchain network operate honestly, while potential malicious entities may attempt to compromise IoT devices, intercept communications, or manipulate credential data. Adversaries may include external attackers, compromised IoT devices (e.g., ESP32 node), or colluding devices and verifiers. We adopt a known-verifiable-credential model, where adversaries attempt to gain sensitive identity information or forge credentials from the system. The adversary capabilities and corresponding security protections are summarized in Table 2.

Table 2. Threat model and adversary capabilities.

Adversary	Capabilities	Security Protections
Outsider	Network attacks (DDoS, MITM), eavesdropping on communications	DID keyAgreement encryption, selective disclosure of VC attributes, only registered service endpoints (HTTPS APIs, MQTT brokers, revocation feeds) are discoverable; blockchain immutability ensures tamper-evident records. VC validity period prevents outdated credentials; zk-SNARK proofs ensure accumulator membership; restricted service endpoints limit communication to authorized controllers.
Compromised IoT Device (Holder/Verifier)	Forge or tamper with stored VCs/VPs, reuse expired credentials	Smart contracts enforce issuance rules, signed on-chain service updates, accumulator verification. zk-SNARK proofs and accumulator confirm legitimate credential membership; selective disclosure prevents overexposure; blockchain ensures traceability, auditing, and prevents service tampering.
Compromised Issuer Node	Issue unauthorized VCs, manipulate accumulator updates	
Colluding Devices + Verifiers	Jointly forge credentials or proofs, manipulate VP verification	

4. Results and Discussion

The efficiency of IoT-SBIdM was evaluated by analyzing its on-chain computational performance and cost metrics. Gas cost, execution time, and storage overhead were compared across all system functions to assess feasibility on resource-constrained IoT devices in blockchain environments.

A proof-of-concept (PoC) implementation was developed to evaluate the proposed system. It includes a React-based front end that enables user interaction with the blockchain via the MetaMask extension. Smart contracts were written in Solidity and compiled, deployed, and managed using the Truffle framework. The system was deployed on an Ethereum-compatible test network using a local Ganache environment with fixed gas prices and block parameters for deterministic evaluation. Smart contracts were executed on a single validating node under a Byzantium-compatible EVM supporting BN254 precompiles for elliptic curve accumulators and zk-SNARK verification.

4.1. Gas Cost of IoT-SBIdM Smart Contracts

Table 3 summarizes the gas consumption of key smart-contract operations within the proposed IoT-SBIdM framework. Gas cost represents the computational effort required to execute blockchain transactions and directly correlates to the operational expense on the Ethereum network. From the results, the `VCRegistry (registerVC)` function incurs the highest cost (170,966 gas), as it involves multi-signature verification and on-chain credential storage. The `VCAccumulator (revoke)` function shows the lowest consumption (23,808 gas), indicating the efficiency of the credential revocation mechanism. In the first experiment, the `SchemaRegistry` contract's gas consumption was 333,583. After refining the contract logic, it decreased to 123,336, achieving an approximate 63% reduction per transaction through targeted optimizations. The initial implementation relied on string-based mapping keys, multiple redundant state variables, and unnecessary storage writes, which contributed to higher gas usage. These were replaced with bytes32 hashes as mapping keys to reduce storage and comparison costs, and redundant state variables and storage operations were removed. The reduction is primarily attributed to the lower computational and storage overhead from hashed keys and streamlined state management. Overall, gas expenditure remains reasonable and acceptable for Industrial IoT (IIoT) applications, demonstrating that IoT-SBIdM achieves a balanced trade-off between security, verifiability, and cost efficiency. The use of off-chain credential storage (via IPFS) and stateless verification design significantly contributes to the reduced on-chain computation and transaction footprint, thereby enhancing scalability. Off-chain scalability is achieved using IPFS as a verifiable data registry. DID documents, JSON-LD contexts, credential schemas, and full verifiable credentials are serialized, pinned, and referenced on-chain only via CIDs or hashes.

Table 3. Gas Costs of IoT-SBIdM Smart Contract Functions.

Smart Contract (Function)	Gas Cost
DIDCIDRegistry ("registerDID")	120,720
DIDCIDRegistry ("updateDID")	46,534
VCAccumulator ("Revoke")	23,808
VCAccumulator ("AddVC")	65,990
VCRegistry ("RegisterVC")	170,966
SchemaRegistry ("registerSchema") (before/after enhancement)	333,583/123,336 (63% cost reduction)

4.2. Blockchain Growth Trend Comparison

To evaluate scalability, the blockchain growth trends of IoT-SBIdM (ours) were compared with related models such as LSTIDM-SB [16], PBIdm [12], and BASS [15] across varying block heights. A comparison chart was created as shown in Figure 7, which presents a comparison of block height and blockchain size over time. The x-axis displays Block Height ($\times 10^3$, ranging from 0 to 5), while the y-axis shows blockchain size (MB, ranging from 0 to 800 in increments of 100). The results demonstrate that our system's

(IoT-SBIdM) relative storage expansion and block size required are 9, 18, 27, 36, and 45 MB for 1000, 2000, 3000, 4000, and 5000 blocks, respectively. At 5000 blocks, IoT-SBIdM requires only 45 MB of storage, resulting in a 55% reduction in storage size compared to [16] (100 MB) and a roughly 94% reduction compared to [12,15] (800 MB) at the same block height.

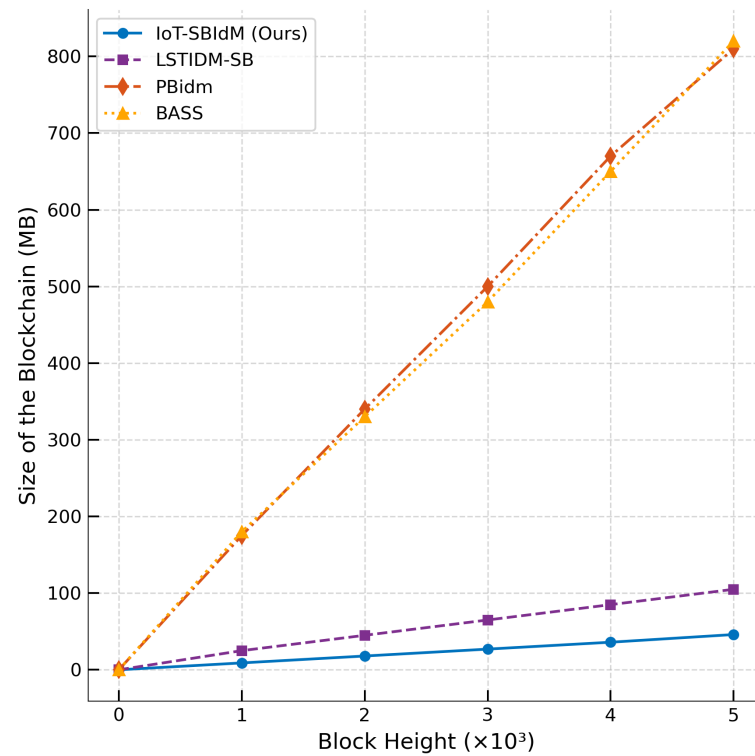


Figure 7. Comparison of blockchain growth trends among IoT-SBIdM, LSTIDM-SB [16], PBIdm [12], and BASS [15].

4.3. Performance Evaluation

Table 4 shows the execution time of the major phases of the IoT-SBIdM workflow measured on two platforms: a standard laptop (representing a typical edge or fog computing environment) and the ESP32 PLUS development board (representing a constrained IoT device). These measurements reflect the cost of executing core identity management operations under identical system configurations. Overall, the results show that IoT-SBIdM maintains practical performance on both resource-rich and resource-constrained platforms, demonstrating its suitability for deployment across heterogeneous IoT environments. The ESP32 PLUS board, despite its limited CPU and memory resources, completes all operations within acceptable latency bounds for real-time IoT authentication and device admission.

Table 4. Execution Times of different Phases of IoT-SBIdM System on Laptop and ESP32 PLUS board.

Phase	Laptop	ESP32 PLUS Board
SystemKeyGen	12.43 ms	10.25 ms
DID Registration	61.83 ms	123.13 ms
CredIssue	3.09 s	4.66 s
CredPresntGen	152.64 ms	114.32 ms
CredVerify	109.10 ms	88.20 ms
CredRevoke	132.15 ms	106.34 ms

The SystemKeyGen phase is lightweight on both platforms, requiring only 12.43 ms on the laptop and 10.25 ms on the ESP32. The lower ESP32 time is attributed to hardware-accelerated cryptographic primitives available on the ESP32's System-on-Chip (SoC), enabling efficient generation of ECDSA key pairs.

DID Registration, which requires storing the DID Document in IPFS and committing its CID to the blockchain, shows a clear difference between platforms. The ESP32 produces a higher latency (123.13 ms) due to network overhead and reduced processing throughput, but the total time remains well below one second, making decentralized identity bootstrapping feasible for low-power devices.

The most time-consuming operation is Credential Issuance (CredIssue), which involves constructing the VC, embedding metadata, and generating a DataIntegrityProof. The laptop achieves issuance in 3.09 s, while the ESP32 requires 4.66 s. This performance gap reflects two factors:

- The computational cost of JSON-LD canonicalization and signature generation.
- Limited RAM and CPU resources on the microcontroller, which impact the normalization of VC structures.

Despite this, issuance still occurs within a small number of seconds, which is acceptable since credential issuance is infrequent. For runtime authentication, Credential Presentation Generation (CredPresentGen) and Credential Verification (CredVerify) are far more critical. Notably, both operations complete in under 160 ms on the laptop and under 120 ms on the ESP32. The ESP32 even outperforms the laptop in CredPresentGen and CredVerify, again due to optimized cryptographic hardware and minimal overhead in its execution environment. This result highlights the system's efficiency for frequent authentication cycles in IoT deployments.

Finally, Credential Revocation (CredRevoke) also shows moderate execution time, taking 132.15 ms on the laptop and 106.34 ms on the ESP32. This indicates that revocation transactions, despite requiring blockchain updates, remain lightweight within the IoT-SBIdM design.

Collectively, these results demonstrate that IoT-SBIdM maintains efficient and scalable execution performance across both high-performance and deeply constrained hardware platforms. This confirms its suitability for real-time use cases such as secure device onboarding, mutual authentication, remote device attestation, and capability verification within IoT networks.

In Figure 8, the execution performance of IoT-SBIdM, BASS, and PBidm is compared across the major operational phases. In the SystemKeyGen phase, IoT-SBIdM completes within 10.25 ms, which is considerably faster than [12,15], which take 497.93 ms and 18.40 ms, respectively, due to its lightweight elliptic-curve generation and hardware-assisted randomness. In the CredIssue phase, IoT-SBIdM records 4.66 s, higher than [12,15], because of additional on-chain anchoring, IPFS interaction, and ECDH-ES/AES 256 GCM envelope encryption on the request/response. These extra steps increase latency; however, they strengthen verifiability and data integrity. For CredProve, IoT-SBIdM requires only 30.24 ms, which is substantially lower than that of [12,15], which take 1.17 s and 104.32 ms, respectively. In the CredVerify phase, IoT-SBIdM completes in 88.20 ms, outperforming [15] (1.31 s) and demonstrating comparable performance to [12] (91.30 ms), benefiting from stateless contract design and optimized elliptic-curve verification. During CredRevoke, IoT-SBIdM records 106.34 ms, faster than [15] (420.30 ms) but slightly higher than [12] (5.21 ms), due to the inclusion of on-chain revocation transactions that ensure verifiable credential invalidation. Overall, IoT-SBIdM achieves lower latency in Key generation, Proof, Verification, and Revocation phases, while maintaining acceptable overhead during Issuance.

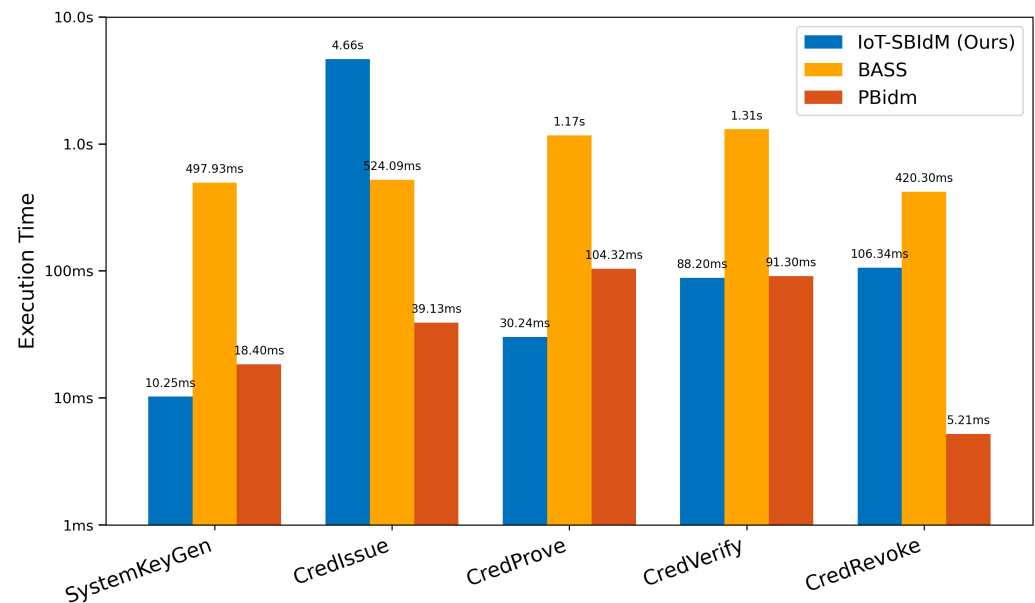


Figure 8. Comparison of Execution Time Across IoT-SBIdM, BASS [15], and PBIdm [12] Systems.

4.4. Practical Deployment Challenges and Limitations

While IoT-SBIdM performs efficiently on both high-performance and constrained IoT devices, practical deployment challenges remain, including heterogeneous device capabilities, network latency, potential blockchain costs for large-scale deployments, and secure key management. Off-chain storage and IPFS interactions may also introduce variability depending on network conditions, which should be considered in real-world deployments.

5. Conclusions

By combining elliptical curve accumulators, zero-knowledge proofs, and W3C-compliant decentralized identity and verifiable credentials data model standards, the proposed IoT-SBIdM framework has efficiently proved its capability to provide a scalable, lightweight, and privacy-preserving identity management solution in IoT scenarios. Our framework achieves superior scalability and storage efficiency by maintaining a compact block size of 45 MB, outperforming comparable schemes with a 55% storage reduction compared to recent models and approximately 94% compared to older systems. Similarly, the efficiency of key generation, proof generation, verification, and revocation in the system was experimentally verified; as well as the fact that credential issuance remains to be the most costly part due to on-chain anchoring, off-chain (e.g., IPFS) interactions, and encryption overhead. Even so, the system performed in real-time on small IoT hardware such as the ESP32 board, proving its adequacy and agreement with study objectives. The results verify that IoT-SBIdM provides a balanced trade-off between computational efficiency, privacy-preserving security, and verifiable credential management for real-world IoT deployments.

Looking further ahead, we could further improve the setup by testing these thresholds on more powerful edge hardware, such as a Raspberry Pi, as well as optimizing the algorithm for issuing credentials, as it exhibited the largest delay (approximately four seconds) when compared to other operations performed in milliseconds. Additionally, the system could be evaluated on IoT devices that are compatible with 5 GHz networks in an attempt to decrease latency while also reducing communication noise. Moreover, long-duration testnet experiments with dynamic credential lifecycle changes and revocation will further validate blockchain storage growth under real-world conditions. Taken together, these

approaches present promising opportunities to enhance the system durability, efficiency, and flexibility in next-generation large-scale IoT deployments.

Author Contributions: Conceptualization, E.A. and U.A.; Methodology, E.A.; Software, E.A.; Formal analysis, E.A.; Writing—original draft, E.A., A.A. and D.A.; Writing—review and editing, A.A., D.A. and U.A.; Visualization, E.A.; Supervision, U.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by a grant (No. CRPG-25-1056) under the Cybersecurity Research and Innovation Pioneers Initiative, provided by the National Cybersecurity Authority (NCA) in the Kingdom of Saudi Arabia.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets used and/or analyzed during the current study are available from the corresponding authors upon request. The data are not publicly available due to the security-sensitive nature of the system, privacy, and protection of cryptographic materials.

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

References

1. Cremonese, B.; Vieira, A.B.; Nacif, J.; Silva, E.F.; Nogueira, M. Identity management for Internet of Things: Concepts, challenges and opportunities. *Comput. Commun.* **2024**, *224*, 72–94. [CrossRef]
2. Ferrag, M.A.; Maglaras, L.A.; Janicke, H.; Jiang, J. Authentication Protocols for Internet of Things: A Comprehensive Survey. *arXiv* **2016**, arXiv:1612.07206. [CrossRef]
3. Bian, Y.; Wang, X.; Jin, J.; Jiao, Z.; Duan, S. Flexible and Scalable Decentralized Identity Management for Industrial Internet of Things. *IEEE Internet Things J.* **2024**, *11*, 27058–27072. [CrossRef]
4. Öztürk, S.B.; Aydos, M. A Blockchain Based Decentralized Identity, Access Management, and Trust Evaluation Framework for IoT. In Proceedings of the 2023 16th International Conference on Information Security and Cryptology (ISCTürkiye), Ankara, Türkiye, 18–19 October 2023. [CrossRef]
5. Ramírez-Gordillo, T.; Maciá-Lillo, A.; Pujol, F.A.; García-D’Urso, N.; Azorín-López, J.; Mora, H. Decentralized Identity Management for Internet of Things (IoT) Devices Using IOTA Blockchain Technology. *Future Internet* **2025**, *17*, 49. [CrossRef]
6. Zhu, X.; Badr, Y. Identity Management Systems for the Internet of Things: A Survey Towards Blockchain Solutions. *Sensors* **2018**, *18*, 4215. [CrossRef] [PubMed]
7. Zhu, X.; Badr, Y. A Survey on Blockchain-Based Identity Management Systems for the Internet of Things. In Proceedings of the 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), Halifax, NS, Canada, 30 July–3 August 2018. [CrossRef]
8. Temoshok, D.; Proud-Madruga, D.; Choong, Y.; Galluzzo, R.; Gupta, S.; LaSalle, C.; Lefkovitz, N.; Regenscheid, A. SP 800-63-4, Digital Identity Guidelines. 2025. Available online: <https://csrc.nist.gov/pubs/sp/800/63/4/final> (accessed on 27 November 2025).
9. Mukhandi, M.; Damião, F.; Granjal, J.; Vilela, J.P. Blockchain-based Device Identity Management with Consensus Authentication for IoT Devices. In Proceedings of the 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 8–11 January 2022. [CrossRef]
10. Huo, R.; Zeng, S.; Wang, Z.; Shang, J.; Chen, W.; Huang, T.; Wang, S.; Yu, F.R.; Liu, Y. A Comprehensive Survey on Blockchain in Industrial Internet of Things: Motivations, Research Progresses, and Future Challenges. *IEEE Commun. Surv. Tutor.* **2022**, *24*, 88–122. [CrossRef]
11. Maram, D.; Malvai, H.; Zhang, F.; Jean-Louis, N.; Frolov, A.; Kell, T.; Lobban, T.; Moy, C.; Juels, A.; Miller, A. CanDID: Can-Do Decentralized Identity with Legacy Compatibility, Sybil-Resistance, and Accountability. In Proceedings of the 2021 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 24–27 May 2021. [CrossRef]
12. Bao, Z.; He, D.; Khan, M.K.; Luo, M.; Xie, Q. PBidm: Privacy-Preserving Blockchain-Based Identity Management System for Industrial Internet of Things. *IEEE Trans. Ind. Inform.* **2023**, *19*, 1524–1534. [CrossRef]
13. Khayer, B.; Mirzaei, S.; Alavizadeh, H.; Salehi Shahraki, A. Blockchain for Secure IoT: A Review of Identity Management, Access Control, and Trust Mechanisms. *IoT* **2025**, *6*, 65. [CrossRef]
14. Sun, X.; Yu, F.R.; Zhang, P.; Sun, Z.; Xie, W.; Peng, X. A Survey on Zero-Knowledge Proof in Blockchain. *IEEE Netw.* **2021**, *35*, 198–205. [CrossRef]

15. Yu, Y.; Zhao, Y.; Li, Y.; Du, X.; Wang, L.; Guizani, M. Blockchain-Based Anonymous Authentication with Selective Revocation for Smart Industrial Applications. *IEEE Trans. Ind. Inform.* **2020**, *16*, 3290–3300. [\[CrossRef\]](#)
16. Zhang, K.; Lee, C.K.M.; Tsang, Y.P. Stateless Blockchain-Based Lightweight Identity Management Architecture for Industrial IoT Applications. *IEEE Trans. Ind. Inform.* **2024**, *20*, 8394–8405. [\[CrossRef\]](#)
17. Ramezan, G.; Meamari, E. zk-IoT: Securing the Internet of Things with Zero-Knowledge Proofs on Blockchain Platforms. *arXiv* **2024**, arXiv:2402.08322.
18. Hu, Q.; Dai, Y.; Li, S.; Jiang, T. Enhancing Account Privacy in Blockchain-Based IoT Access Control via Zero Knowledge Proof. *IEEE Netw.* **2023**, *37*, 117–123. [\[CrossRef\]](#)
19. Sporny, M.; Longley, D.; Chadwick, D.; Herman, I. Verifiable Credentials Data Model v2.0. Available online: <https://w3c.github.io/vc-data-model/> (accessed on 27 November 2025).
20. Sporny, M.; Longley, D.; Sabadello, M.; Reed, D.; Steele, O.; Allen, C. Decentralized Identifiers (DIDs) v1.0. Available online: <https://www.w3.org/TR/did-1.0/> (accessed on 27 November 2025).
21. Hsieh, Y.H.; Tseng, C.H.; Luo, B.Y.; Yuan, S.M. Cross-Border Digital Identity System Based on Ethereum Layer 2 Architecture. *Electronics* **2026**, *15*, 708. [\[CrossRef\]](#)
22. Li, X.; Chen, C.; Teng, M.; Shi, Y. Blockchain-based Traceable Selective Disclosure Credentials for Self-Sovereign Identity. In Proceedings of the 2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Tianjin, China, 8–10 May 2024. [\[CrossRef\]](#)
23. Ra, G.; Kim, T.; Lee, I. VAIM: Verifiable Anonymous Identity Management for Human-Centric Security and Privacy in the Internet of Things. *IEEE Access* **2021**, *9*, 75945–75960. [\[CrossRef\]](#)
24. Deebak, B.D.; Memon, F.H.; Dev, K.; Khowaja, S.A.; Wang, W.; Qureshi, N.M.F. TAB-SAPP: A Trust-Aware Blockchain-Based Seamless Authentication for Massive IoT-Enabled Industrial Applications. *IEEE Trans. Ind. Inform.* **2023**, *19*, 243–250. [\[CrossRef\]](#)
25. Sonnino, A.; Al-Bassam, M.; Bano, S.; Meiklejohn, S.; Danezis, G. Coconut: Threshold Issuance Selective Disclosure Credentials with Applications to Distributed Ledgers. *arXiv* **2020**, arXiv:1802.07344. [\[CrossRef\]](#)
26. Luong, D.A.; Park, J.H. Privacy-Preserving Identity Management System on Blockchain Using Zk-SNARK. *IEEE Access* **2023**, *11*, 1840–1853. [\[CrossRef\]](#)
27. Longley, D.; Sporny, M.; Bernstein, G. Data Integrity ECDSA Cryptosuites v1.0. Available online: <https://www.w3.org/TR/vc-di-ecdsa/> (accessed on 27 November 2025).
28. Benet, J. IPFS-Content Addressed, Versioned, P2P File System. *arXiv* **2014**, arXiv:1407.3561. [\[CrossRef\]](#)
29. Yasuda, K.; Jones, M. Self-Issued OpenID Provider v2. Available online: https://openid.net/specs/openid-connect-self-issued-v2-1_0-ID1.html (accessed on 28 November 2025).
30. Terbu, O.; Lodderstedt, T.; Yasuda, K.; Fett, D.; Heenan, J. OpenID for Verifiable Presentations 1.0. Available online: https://openid.net/specs/openid-4-verifiable-presentations-1_0-final.html (accessed on 28 November 2025).
31. IEEE Std 802.11be-2024 (Wi-Fi 7); IEEE Standard for Information Technology Telecommunications. IEEE: Piscataway, NJ, USA, 2024.
32. Breu, R.; Hinkel, U.; Hofmann, C.; Klein, C.; Paech, B.; Rumpe, B.; Thurner, V. Towards a formalization of the Unified Modeling Language. In *ECOOP'97—Object-Oriented Programming*; Springer: Berlin/Heidelberg, Germany, 1997; pp. 344–366. [\[CrossRef\]](#)

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.