

Article

Decentralized Authentication and Data Access Control Scheme Using DID for Fog-Enabled Industrial Internet of Things

Kisung Park 

Department of Smart Security, Gachon University, Seongnam 13120, Republic of Korea; kisung@gachon.ac.kr

Abstract

The Industrial Internet of Things (IIoT) integrates a wide range of devices and identities, making the protection of sensitive industrial data a critical challenge. However, existing centralized systems still face limitations such as single points of failure, inefficient identity authentication, and dependence on trusted third parties (TTPs). To address these issues, we present a blockchain-based authentication and data access control scheme for IIoT systems. The proposed scheme eliminates TTP involvement by employing decentralized identifiers (DIDs) and key-aggregate searchable encryption (KASE), utilizing scalable authentication without requiring all industrial data to be stored on the blockchain. Security robustness is demonstrated through informal analysis, the Real-or-Random (ROR) model, and the AVISPA simulation tool (v1.6). Furthermore, performance evaluation using the Multiprecision Integer and Rational Arithmetic Cryptographic Library (MIRACL) SDK shows that the proposed scheme achieves computational efficiency compared with existing solutions. Overall, the results confirm that the proposed scheme provides secure, efficient, scalable, and TTP-free data management for IIoT environments.

Keywords: authentication; access control; decentralized identifiers; security analysis; blockchain**MSC:** 68-XX; 94Axx; 94A62; 68Mxx

Academic Editor: Cheng-Chi Lee

Received: 14 October 2025

Revised: 11 November 2025

Accepted: 12 November 2025

Published: 17 November 2025

Citation: Park, K. Decentralized Authentication and Data Access Control Scheme Using DID for Fog-Enabled Industrial Internet of Things. *Mathematics* **2025**, *13*, 3686. <https://doi.org/10.3390/math13223686>

Copyright: © 2025 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The Industrial Internet of Things (IIoT) constitutes a large-scale ecosystem in which heterogeneous devices interconnect to exchange data and collaborate for intelligent decision-making. It was projected that by 2020, more than 25 billion smart devices would be deployed worldwide [1]. Such rapid growth, however, imposes substantial burdens on existing infrastructures, particularly in terms of computation, storage, and security. To mitigate these limitations, the integration of IIoT with cloud computing has been extensively explored, leveraging the cloud's virtually unlimited computational and storage resources.

Cloud-assisted IIoT systems are envisioned to underpin critical infrastructures such as smart manufacturing, intelligent transportation, and advanced energy systems. These systems depend on massive volumes of industrial data to enable analytics-driven operations. However, the geographical separation between centralized cloud servers (CSs) and edge devices introduces latency, limiting their effectiveness in delay-sensitive and mobile applications. Fog computing has thus been introduced as a complementary paradigm that extends cloud capabilities to the network edge [2]. By deploying fog nodes between CSs and end devices, data can be preprocessed and temporarily stored closer to its source, thereby reducing latency and improving overall system efficiency compared with the traditional device–cloud model.

In cloud-assisted IIoT systems, data sharing typically involves multiple categories of industrial information distributed among various organizations and individuals. For example, a smart factory gathers production, environmental, and energy data through gateway nodes and grants selective access to external parties. Specifically, production and energy data may be shared with the energy management provider, whereas production and environmental data may be disclosed to the quality inspection department. While such fine-grained sharing enhances collaboration and decision-making, it simultaneously heightens the risk of data breaches. To address these concerns, searchable encryption (SE) [3,4] has been adopted to enable secure keyword searches over encrypted datasets [5–8]. Nonetheless, managing multiple data categories across different entities requires numerous encryption keys, resulting in high communication and storage overhead for secure distribution and maintenance. To alleviate this issue, key-aggregate searchable encryption (KASE) [9] has been introduced and further advanced in subsequent studies [10,11]. As illustrated in Figure 1, with KASE, a factory can distribute a single aggregate key to each authorized user, thereby streamlining key management and enabling a single trapdoor to retrieve multiple classes of data.

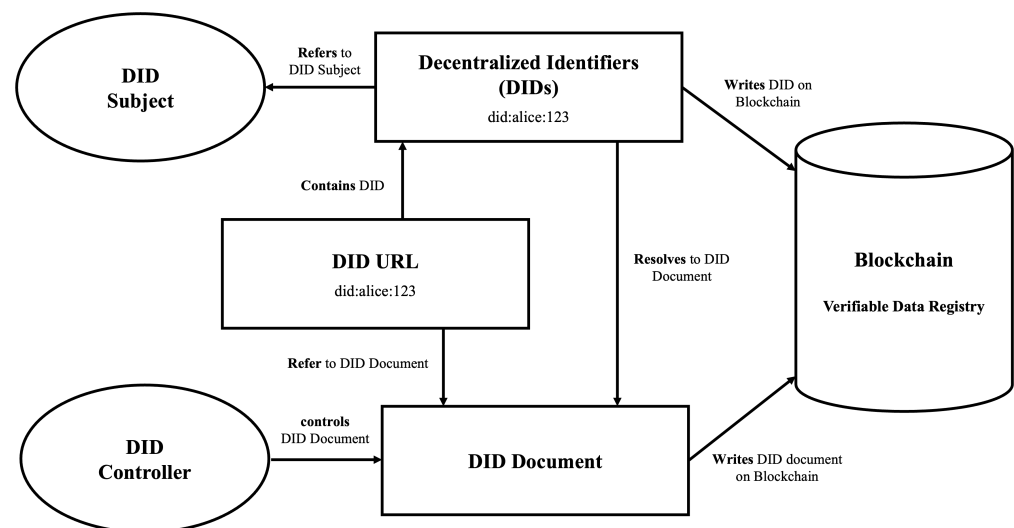


Figure 1. DID architecture.

In parallel, blockchain technology has emerged as a promising solution due to its decentralization, immutability, and capability to enable trustless interactions without reliance on trusted third parties (TTPs) [12]. Nonetheless, cloud-assisted infrastructures remain vulnerable to adversarial attacks seeking to exploit inherent weaknesses [13]. Furthermore, most existing protocols continue to rely on trusted authorities, reintroducing centralization and associated risks such as performance bottlenecks, privacy leakage, and single points of failure. In cloud-assisted IIoT, these dependencies extend beyond data management to include user identity and access rights, which are often still governed by TTPs.

To overcome these challenges, recent studies have explored decentralized schemes that eliminate reliance on TTPs. For example, attribute-based encryption (ABE) [14] combined with blockchain supports fine-grained access control. However, such schemes remain only partially decentralized, as they still require a TTP for private key issuance and policy management, thereby exposing systems to insider threats and privacy risks. More recently, decentralized identifiers (DIDs) [15] have been proposed as a promising alternative for privacy-preserving identity management. DIDs enable devices to autonomously generate and manage their identifiers and cryptographic keys without TTP involvement, thus achieving verifiable and self-sovereign identity in distributed settings.

In this work, we propose a blockchain-enabled authentication and access control framework tailored for IIoT environments. The scheme integrates DIDs and KASE to eliminate reliance on centralized authorities for identity management, enhance self-sovereign identity, and achieve scalable, fine-grained access control. The proposed design ensures secure, efficient, and resilient authentication, thereby addressing the stringent requirements of next-generation industrial infrastructures.

1.1. Motivations

Cloud/Fog-assisted IIoT environments must support secure and efficient sharing of heterogeneous industrial data. Existing solutions introduce significant key management overhead and still rely on TTPs. This reliance creates centralization risks, privacy leakage, and single points of failure. Blockchain-based schemes attempt to reduce trust dependence, but many of them keep identity issuance and authentication under centralized control. In addition, these schemes often utilize static identity information, which increases exposure to linkability attacks and replay attacks. Furthermore, current data access control mechanisms are often insufficient for multi-category industrial data, and many existing models lack the ability to support fine-grained access policies across diverse entities in IIoT settings. These issues demonstrate the need for a decentralized authentication and access control scheme that can simplify key management for multi-category data, provide self-sovereign and privacy-preserving identity control without TTPs, and support fine-grained and flexible access control. Motivated by these requirements, this work combines DIDs and KASE within a blockchain-supported scheme to offer scalable, resilient, and fine-grained authentication and access control in Fog-assisted IIoT systems.

1.2. Contributions

The main contributions of this work can be summarized as follows:

- We present a decentralized protocol for authentication and access control in cloud-assisted IIoT environments. To guarantee scalability and strong security for participating nodes, the scheme employs DIDs and KASE. Our scheme addresses the challenges of TTP dependence, achieves efficient key management, and supports fine-grained access control in industrial data sharing.
- Within the proposed scheme, the data owner, data user, and fog node reveal only a minimal portion of personal information when exchanging messages over public networks. Each entity is capable of independently creating and maintaining its identity, along with corresponding public and private keys. Importantly, during the authentication process among the data owner, data user, and fog node, all session-related parameters are renewed in every session, with the exception of the user's DID. These features mitigate the vulnerabilities associated with static identifiers and reduce exposure to linkability and replay attacks.
- The security guarantees of the proposed scheme are examined through a series of evaluations, including informal analysis, the Real-or-Random (ROR) model [16], and formal verification conducted using the Automated Validation of Internet Security Protocols and Applications (AVISPA) tool [17]. These analyses demonstrate that the protocol remains secure even without any TTP involvement and confirm its robustness against a wide range of attacks.
- To analyze performance of our scheme, we measure and compare the computational costs of the proposed protocol against existing schemes. The cryptographic operations are implemented and tested using the Multiprecision Integer and Rational Arithmetic Cryptographic Library (MIRACL) SDK [18]. The results demonstrate that the protocol

maintains practical efficiency while providing enhanced decentralization and stronger protection against the potential attacks.

1.3. Organization

The remainder of this paper is structured as follows. Section 2 reviews the existing state-of-the-art solutions. The preliminaries and threat model are discussed in Section 3. Section 4 details the proposed scheme, followed by a comprehensive security analysis in Section 5. Section 6 presents experimental results for cryptographic primitives, and compares the proposed scheme with related works. Finally, Section 7 concludes the paper.

2. Related Works

Over the past decade, a wide range of privacy-preserving schemes have been introduced to safeguard user data and privacy within IoT and IIoT ecosystems. Shuai et al. [19] presented a mechanism that protects sensitive data collected from industrial sensors against unauthorized access. Their approach employs Rabin cryptography as the core primitive and integrates password verification tables, aiming to balance security with computational efficiency. Srinivas et al. [20] proposed a lightweight solution supported by both formal and informal security analyses. Their system incorporates a fuzzy extractor for biometric verification, with performance validated through NS2 simulation. This design facilitates interoperability and provides users with remote control of smart home devices via Internet-enabled mobile platforms. Wazid et al. [21] designed a three-factor authentication protocol tailored for smart home environments. A registration server is utilized to manage entity enrollment and key distribution, functioning as a central server in cloud-based IoT. Their model supports authentication among users, gateways, IoT devices, and cloud servers, although the registration server primarily acts as a registrar.

Banerjee et al. [22] introduced an ABE system for privacy-preserving data protection, while Mubarakali [23] proposed a blockchain-enabled system for personal health record sharing, using attribute-based tokens to regulate access among healthcare and insurance entities. Despite their privacy benefits, both approaches depend on centralized infrastructures, which remain susceptible to single points of failure. Park and Park [24] proposed a blockchain-based privacy-preserving scheme for IoT environments, validated through testbed implementation and evaluated under the “Dolev–Yao (DY) model” adversarial model [25]. Their framework enables authentication and data sharing between data owners and users without relying on a TTP. However, the scheme employs ABE, which requires re-encryption whenever new ciphertexts are generated, which leads to additional computational overhead. Eddine et al. [26] proposed a blockchain-based security architecture for the Internet of Vehicles. While computational and communication metrics were shown to be efficient, the scheme fails to guarantee anonymity and is vulnerable to attacks such as DoS, insider threats, and device theft. Moreover, it lacks scalability features such as dynamic node addition. Tomar and Tripathi [27] introduced an ECC-based protocol for fog-enabled vehicular networks, which demonstrates low computational and communication costs. However, the scheme does not resist common attacks and similarly lacks support for dynamic scalability. Zhang et al. [28] combined ciphertext-policy attribute-based keyword search (CP-ABKS) with blockchain to provide searchable encryption, integrity verification, and auditing. Despite these advantages, ABE-based solutions often encounter challenges with re-encryption overhead and key management, particularly in large-scale deployments. Jia et al. [29] proposed an ECC-based protocol for IoT healthcare applications that ensures anonymity and untraceability but remains vulnerable to insider, DoS, and offline guessing attacks, with no scalability support. Karankar and Seth [30] proposed an ABE-based access

control scheme for IoT. However, their scheme also suffers from the drawback that the entire ciphertext must be re-encrypted whenever the access policy is updated.

Niu et al. [10] introduced a KASE-based blockchain scheme for PHR sharing, yet it lacked secure mutual authentication and multi-keyword search. More recently, Trivedi and Patel [31] developed a KASE-based protocol for distributed IoT healthcare, where only authorized medical data users could query encrypted data using attribute tokens. Guo et al. [32] presented a lightweight remote authentication protocol using pseudo-identities to ensure anonymity in IoT environments. However, their scheme does not incorporate blockchain features and lacks essential functionalities such as dynamic node addition and password update. Vangala et al. [33] designed a blockchain-based authenticated key agreement protocol for precision agriculture, which proved resilient against several attacks in real-world evaluations, though scalability remains limited. Likewise, Ponnuru et al. [34] proposed a blockchain-oriented authentication mechanism for fog-assisted IIoT, but their solution still depends on a trusted third party to manage user identities and private keys. Lee et al. [35] proposed a blockchain-based KASE scheme for medical IoT to support multi-delegation. However, their scheme does not support decentralized identity and is not suitable for deployment in fog computing environments.

A summary of existing authentication and data access schemes for cloud/fog-assisted IIoT environments is provided in Table 1.

Table 1. Comparison of previous schemes in IIoT environments.

Scheme	Year	Cryptographic Primitives	Advantages/Description	Shortcomings/Limitations
[32]	2024	* Secure hash function	* Blockchain-based lightweight authentication scheme for fog-enabled IoT * Provides password update phase	* Does not provide decentralized identity * Does not support fin-grained access control * Does not achieve self-sovereignty * Does not perform formal simulation analysis
[33]	2023	* Elliptic curve cryptography * Secure hash function	* Blockchain-based authentication scheme for fog-enabled IoT * Provides blockchain simulation * Provides formal security analysis	* Does not provide decentralized identity * Does not support fin-grained access control * Does not achieve self-sovereignty
[34]	2025	* Elliptic curve cryptography * Secure hash function	* Blockchain-assisted authentication scheme for fog-enabled IoT * Involves four entities in authentication phase * Provides formal security analysis	* Does not support decentralized identity * Does not support fin-grained access control * Does not achieve self-sovereignty
[10]	2020	* Bilinear Pairing * KASE * Secure hash function	* Blockchain-based KASE scheme for IoT * Provides fine-grained access control	* Does not support decentralized identity * Dose provide formal simulation analysis
[30]	2025	* Bilinear Pairing * ABE * Secure hash function	* An IoT system for access control uisng blockchain * Provides blockchain simulation	* Does not support decentralized identity * Does not perform formal security analysis * Does not achieve self-sovereignty

Table 1. Cont.

Scheme	Year	Cryptographic Primitives	Advantages/Description	Shortcomings/Limitations
[35]	2024	* Bilinear Pairing * KASE * Secure hash function	* Blockchain-enabled KASE for medical IoT * Provides fine-grained access control	* Does not support decentralized identity * Dose provide formal simulation analysis

3. Preliminaries

3.1. Decentralized Identifiers

Decentralized Identifiers (DIDs) [15] serve as unique identifiers for entities by leveraging distributed ledger technologies, without reliance on centralized authorities. Within the DID ecosystem, three key roles exist: issuers (e.g., government agencies) that generate verifiable credentials, holders who maintain these credentials in digital wallets, and verifiers responsible for validating their authenticity. The interactions among these roles and the corresponding information flows are illustrated in Figure 1.

As shown in Figure 1, the DID architecture is built upon DID Documents [15], which contain cryptographic materials, verification methods, and service endpoints used to interact with the entity represented by the DID. A DID consists of a method and a method-specific identifier that designates the DID subject, which may be a person, device, organization, or software system. The DID controller is the entity authorized to modify the DID Document, typically through possession of the relevant cryptographic keys, and the controller may or may not be identical to the DID subject. A DID URL extends the basic DID syntax by including path, query, or fragment components, enabling the precise identification of specific resources—such as a public key or service endpoint—either within the DID Document or external to it. DID Methods define the procedures for creating, resolving, updating, and deactivating DIDs and DID Documents on a verifiable data registry, such as a blockchain or decentralized storage network. DID resolvers use the corresponding DID Method to retrieve DID Documents through DID resolution, while DID URL dereferencers fetch the specific resource indicated by a DID URL. These components also form a unified and interoperable system that supports decentralized, verifiable, and fine-grained identity management across heterogeneous environment.

3.2. Threat Model

We evaluate the security of the proposed scheme using the “Dolev–Yao (DY) model” [25], which is commonly adopted for rigorous protocol analysis. In this setting, all communications are assumed to traverse untrusted channels, giving an adversary extensive control over the network. Under these assumptions, the adversary can monitor, block, or store transmitted messages, as well as manipulate their contents, fabricate new ones, and insert malicious data into ongoing exchanges.

3.3. Mathematical Preliminaries

We have outlined several mathematical definitions and cryptographic primitives to facilitate readers’ understanding of the proposed approach. The scheme is constructed on elliptic curve cryptography (ECC) and its security relies on the hardness of the elliptic curve discrete logarithm problem (ECDLP) and the elliptic curve decisional Diffie–Hellman problem (ECDDHP). Utilizing these computational assumptions, we provide a formal analysis to demonstrate the security of the proposed scheme.

Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) is a public-key cryptographic scheme that employs the algebraic properties of elliptic curves to achieve security. To apply ECC in a security setting, one must first choose a large finite field $\mathbb{F}_p$, a large prime p , another prime q , and elliptic curve parameters w and v . Based on these parameters, an elliptic curve is defined as $E(w, v) : y^2 = x^3 + wx + v$, subject to the condition $4w^3 + 27v^2 \neq 0$. The points on this curve form an additive group, for which a base point P is specified. For an integer $n \in \mathbb{Z}_q$, scalar multiplication is expressed as:

$$n \cdot P = \underbrace{P + P + \cdots + P}_{n \text{ times}}$$

The cryptographic strength of ECC relies on the intractability of the following problems:

- Elliptic Curve Discrete Logarithm (ECDL) Problem: Determine $n \in \mathbb{Z}_q$ given $n \cdot P$.
- Elliptic Curve Decisional Diffie–Hellman (ECDDH) Problem: Given n, s , and t , decide whether $n \cdot s \cdot P = t \cdot P$.
- Elliptic Curve Computational Diffie–Hellman (ECCDH) Problem: Compute $n \cdot s \cdot P$ given $n \cdot P$ and $s \cdot P$.

3.4. Bilinear Pairing

Bilinear maps constitute a fundamental building block of pairing-based cryptography, which underlies schemes such as identity-based encryption [36] and KASE [9]. Consider two cyclic groups G_1 and G_2 of prime order p , and let g be a generator of G_1 . A bilinear map is defined as a function $e : G_1 \times G_1 \rightarrow G_2$ that satisfies the following conditions:

- (1) Bilinearity: For all $x, y \in \mathbb{Z}p^*$ and $\forall g \in G_1$, $e(g^x, g^y) = e(g, g)^{xy}$.
- (2) Non-degeneracy: $e(g, g) \neq 1$.

When both the group operation in G_1 and the evaluation of e can be performed efficiently, the pair (G_1, G_2) is termed a bilinear group.

4. Proposed Scheme

In this section, we describe the system model and access control workflow of the proposed scheme. The primary objective is to enable patients to manage and share their data securely in a real-time and a self-sovereign manner. IIoT data are stored on fog nodes, while the big data center collects only metadata required for analytical purposes rather than the raw data itself. The notations used throughout this paper are summarized in Table 2.

Table 2. Notations for our scheme.

Symbol	Meaning
DO	Data owner
FN	Fog node
DU	Data user
$DID_{DO}, DID_{DU}, DID_{FN}$	DID of DO , DU , and FN
n, c	Random number and challenge
$e()$	Bilinear pairing
$E(a, b)$	Elliptic curve
$pk_{DO/DU}, sk_{DO/DU}$	Key pair of DO and DU for authentication

Table 2. Cont.

Symbol	Meaning
$pk_{SDO/SDU}, sk_{SDO/SDU}$	Key pair of <i>DO</i> and <i>DU</i> for data sharing
$DDoc_{DO/DU}$	DID documents of <i>DO</i> and <i>DU</i>
$h(\cdot)$	One-way hash function
$ $	Concatenation

4.1. System Model

We begin by outlining the system model, as shown in Figure 2. The architecture is composed of four primary entities: the data user (DU), the data owner (DO), the fog node (FN), and the big data center (BDC).

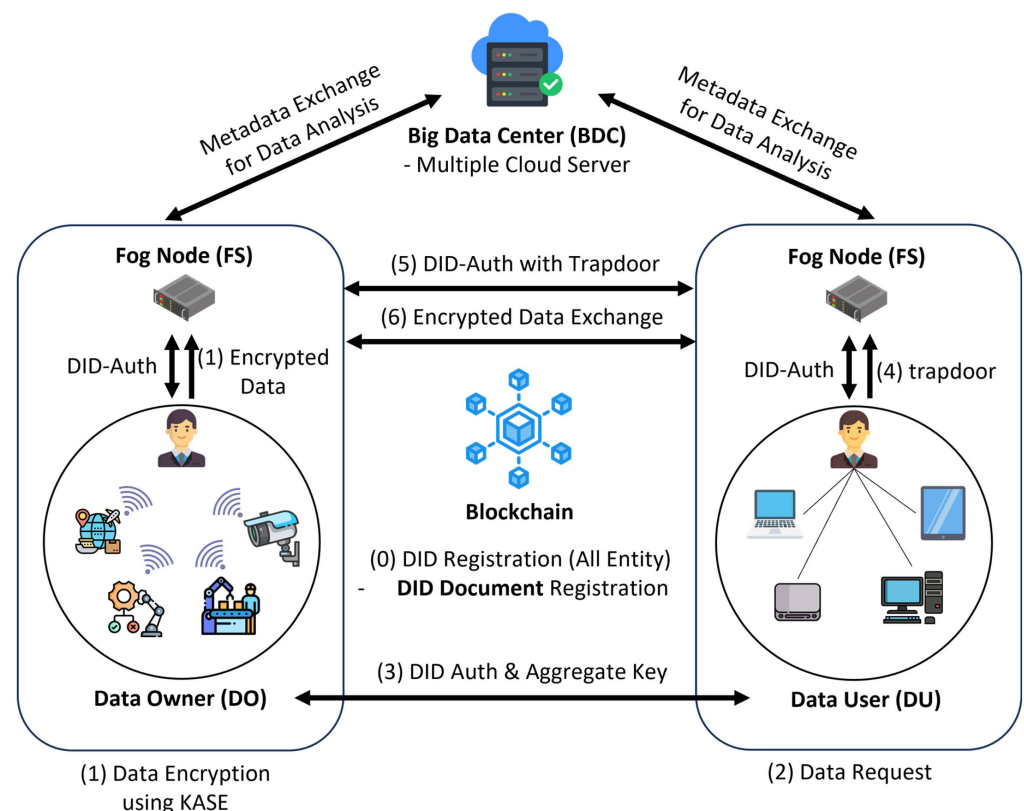


Figure 2. System model.

- **Big data center (BDC):** The BDC is composed of multiple cloud servers and analytical tools, which receive data from fog nodes to perform large-scale processing. Data uploaded from the IoT network is analyzed within the BDC and subsequently utilized to provide requested services to users. This hierarchical architecture supports efficient and secure management of IoT data in fog-enabled systems.
- **Fog node (FN):** Each FN is responsible for managing a single IoT network, while interconnected FNs collectively form a resilient fog-enabled infrastructure. This infrastructure enables secure data processing and facilitates data sharing. Data collected from IoT devices are transmitted to the BDC upon request for advanced services.
- **Data owner (DO):** The DO serves as a domain administrator, responsible for collecting data from IIoT devices deployed in designated areas to capture and deliver real-time information to data users. Each device is associated with a data owner who oversees the IoT network and supports scalability by managing the integration of new devices.

Acting as the IIoT device owner, the *DO* establishes the connection between the IIoT network and the *FN*.

- Data user (*DU*): The *DU* refers to a domain administrator in another network domain who requests access to data managed by the *DO* and utilizes the shared data within their own domain.

The communication flow of the proposed scheme can be described as follows:

- S1: Both the *DO* and the *DU* generate their respective DIDs and perform DID-based authentication with their associated *FN* to initiate IIoT service access. Once the authentication is completed, the *DO* encrypts the IIoT data and uploads it to the *FN*.
- S2: When the *DU* intends to access the IIoT data of the *DO*, it sends a data request message to the *DO* and performs DID authentication. After successful authentication, the *DO* issues and transmits an aggregate key corresponding to the requested data to the *DU*.
- S3: To retrieve the encrypted data, the *DO* generates a trapdoor and forwards it to its *FN*. The *FN* then communicates with the *DO*'s domain *FN* and performs DID authentication. Upon successful verification, the requested encrypted data is transferred to the *DO*'s *FN* and subsequently decrypted. The *DU* then receives the encrypted data, decrypts it using the aggregate key, and utilizes the retrieved information within its own IIoT domain.

4.2. Initialization Phase

In this initialization phase, the *DO*, *DU*, and *FN* generate their respective parameters required for subsequent operations. Once these parameters are produced, they are recorded on the blockchain to enable the next phase. The initialization procedure is identical for the *DO*, *DU*, and *FN*.

- S1: The *DO* begins by constructing a bilinear group (G_1, G_2) of prime order q , an elliptic curve $E(a, b) : y^2 = x^3 + ax + b \pmod{q}$, and defining a non-degenerate bilinear map $e : G_1 \times G_1 \rightarrow G_2$. The *DO* also specifies the maximum number of documents n with indices ind , selects a generator $g \in G_1$, a random secret $\alpha \in \mathbb{Z}_q$, and a cryptographic hash function $h : 0, 1^* \rightarrow \mathbb{Z}_q^*$.
- S2: The *DO* first creates its decentralized identifier DID_{DO} and then computes $g_i = g^{\alpha_i} \in G_1$ for every $1 \leq i \leq 2n$. A polynomial $f(x)$ of degree ind is subsequently constructed. In addition, the *DO* generates two distinct key pairs: $(sk_{DO}, pk_{DO}) = (sk_{DO}, g^{sk_{DO}})$, which is used for authentication, and $(sk_{SDO}, pk_{SDO}) = (sk_{SDO}, g^{sk_{SDO}})$, which is employed for data sharing. Each key pair includes a secret key and its associated public key.
- S3: Finally, the *DO* publishes the document $DDoc_{DO} = \{E(a, b), q, G_1, G_2, e, g, n, pk_{DO/SDO}, ind, (g_i)_{1 \leq i \leq 2n}, h, (g^{f(x)})_{0 \leq x \leq ind}\}$ on the blockchain.

4.3. Registration Phase

To join the proposed network, both the *DO* and *DU* undergo a registration phase by performing DID-based authentication with their respective *FN*s. Through this process, each of them establishes a pre-shared key with the *FN* in its domain. The registration procedure of the *DO* is identical to that of the *DU* and proceeds as follows:

- S1: The *DO* begins the process by generating a random nonce $n_1 \in \mathbb{Z}_q$ and sending a registration request, which includes DID_{DO} , to FN_i through a secure communication channel.

- S2: Upon reception, FN_i resolves the on-chain DID document Doc_{DO} to obtain the public parameters together with pk_{DO} . It then creates a challenge c_1 and transmits it back to the DO via the secure channel.
- S3: Once the challenge is received, the DO computes $C_1^{DO} = n1 \cdot G$ and $C_2^{DO} = n1 + c_1 \cdot sk_{DO}$, and subsequently returns the tuple C_1^{DO}, C_2^{DO} to FN_i .
- S4: FN_i then verifies the response by checking the equality $C_2^{DO} \cdot G - C_1^{DO} \stackrel{?}{=} c_1 \cdot pk_{DO}$. If the condition holds, both entities compute the shared session key, where the DO derives $sk_{DO/FN_i} = pk_{FN_i} \cdot c_1 \cdot sk_{DO}$ and FN_i derives $sk_{FN_i/DO} = pk_{DO} \cdot c_1 \cdot sk_{FN_i}$.

4.4. Data Upload Phase

Once the initialization phase is completed, the DO proceeds to encrypt the data before uploading it to the FN . The detailed steps are as follows:

- S1: The DO chooses a random value $rn_1 \in \mathbb{Z}_q^*$ and generates the public parameters for each index i , represented as $PP_i = (cp_1, cp_2, cp_3)$ for $i = 1, \dots, n$. These values are computed as $cp_1 = g^{rn_1}$, $cp_2 = (g_i \cdot pk_{SDO})^{rn_1}$, and $cp_3 = D \cdot e(g_1, g_n)^{rn_1}$. The data index value is then derived as $INDX_i = \frac{e(g, h_1(ind))^{rn_1}}{e(g_1, g_n)^{rn_1}}$.
- S2: The DO encrypts the data $PP_i, INDX_i$ using the pre-shared session key sk_{DO/FN_i} . Finally, the encrypted data $(PP_i, INDX_i)_{sk_{DO/FN_i}}$ are uploaded by the DO to the FN .

4.5. Data Request and Aggregate Key Sharing Phase

To initiate the data access phase, the DU issues a request to the DO to obtain the aggregate key. Upon confirming the validity of the request, the DO delegates authorization to FN_i to handle subsequent access control procedures. The detailed process of this phase is outlined below.

- S1: The DU begins by generating a random nonce rn_1 and a challenge value c_1 . It then computes $CT_1^{DU} = rn_1 \cdot G$ and transmits the tuple CT_1^{DU}, DID_{DU}, c_1 to the DO .
- S2: Upon receiving the message, the DO selects a random nonce rn_2 and generates a challenge c_2 . It retrieves the public key pk_{DU} of the DU from its DID document stored on the blockchain. Subsequently, the DO computes $CT_1^{DO} = rn_2 \cdot G$, $m_1 = (DID_{DO} || DID_{DU} || c_1 || c_2 || CT_1^{DU} || CT_1^{DO})$, and $CT_2^{DO} = rn_2 + h(m_1) \cdot sk_{DO}$. It then derives the session key $sk_{DO-DU} = rn_2 \cdot CT_1^{DU}$ and computes the aggregate key as $akey = \prod_{j \in D} g_{n+1-j}^{sk_{DMDO}}$. The aggregate key is encrypted using the session key to obtain $Eakey = (akey)_{sk_{DO-DU}}$. Finally, the DO sends $Eakey, c_2, CT_1^{DO}$ to the DU .
- S3: Upon receiving the response, DU retrieves the DO 's public key pk_{DO} from the DID document recorded on the blockchain. It then constructs $m_2 = (DID_{DO} || DID_{DU} || c_1 || c_2 || CT_1^{DU} || CT_1^{DO})$ and verifies the validity of the received message by checking whether $CT_2^{DO} \cdot G - CT_1^{DO} \stackrel{?}{=} h(m_2) \cdot pk_{DO}$. If it is hold, the DU computes the session key as $sk_{DU-DO} = rn_1 \cdot CT_1^{DO}$ and decrypts $Eakey$ using this session key to recover the aggregate key $akey$. Finally, the DU securely stores $akey$ for use in subsequent data access operations.

4.6. Fog-Enabled Data Sharing Phase

The DU initiates a data access request to its domain-specific fog node FN_j . During this process, the DU communicates with FN_j using their established pre-shared key and transmits the corresponding trapdoor. Subsequently, the fog node FN_i in the DO 's domain performs DID-based authentication with FN_j . Upon successful verification, FN_i shares the requested encrypted data with FN_j based on the received trapdoor information.

- S1: DU selects the random number rn_1 and computes $C_1^{DU} = rn_1 \cdot G$, $CT_2^{DU} = rn_1 + h(D_{req}) \cdot sk_{DU}$, and the trapdoor $TD_j = akey \cdot h(ind)$. Subsequently, $\{CT_1^{DU}, CT_2^{DU}, Tr_j\}$ is encrypted using pre-shared key sk_{DU-FN_j} and transmitted to FN_j .
- S2: Upon receiving the message, FN_j decrypts it using sk_{DU-FN_j} to obtain $\{CT_1^{DU}, CT_2^{DU}, Tr_j\}$. It verifies the correctness of the message by checking $CT_2^{DO} \cdot G - CT_1^{DU} \stackrel{?}{=} h(D_{req}) \cdot pk_{DO}$. If the verification succeeds, FN_j generates a random value rn_2 and challenge c_1 , computes $C_1^{FN_j} = rn_2 \cdot G$, and sends $\{CT_1^{FN_j}, DID_{FN_j}, c_1\}$ to FN_i .
- S3: After receiving the request, FN_i generates a random number rn_3 and challenge c_2 , computes $C_1^{FN_i} = rn_3 \cdot G$, $m_1 = (DID_{FN_i} || DID_{FN_j} || c_1 || c_2 || CT_1^{FN_i} || CT_1^{FN_j})$, and $CT_2^{FN_i} = rn_3 + h(m_1) \cdot sk_{FN_i}$. It then derives the session key $sk_{FN_i/j} = rn_3 \cdot CT_1^{FN_j}$. FN_i and transmits $\{CT_1^{FN_i}, CT_1^{FN_j}, DID_{FN_i}, c_2\}$ to FN_j .
- S4: Upon receiving the response, FN_j retrieves the FN_i 's public key pk_{FN_i} from the blockchain and computes $m_2 = m_1 = (DID_{FN_i} || DID_{FN_j} || c_1 || c_2 || CT_1^{FN_i} || CT_1^{FN_j})$. It verifies whether $CT_2^{FN_i} \cdot G - CT_1^{FN_i} \stackrel{?}{=} h(m_2) \cdot pk_{FN_i}$. If valid, it derives the session key $sk_{FN_j/i} = rn_2 \cdot CT_1^{FN_i}$. FN_j encrypts the trapdoor Tr_j using $sk_{FN_j/i}$, and forwards it to FN_i .
- S5: After decrypting Tr_j using $sk_{FN_j/i}$, FN_i computes $ccp_1 = \prod_{k \in S, k \neq i} g_{n+1-k+i}$, $ccp_2 = \prod_{k \in D} g_{n+1-k}$, $ccp_3 = cp_3 \cdot \frac{e(ccp_1, cp_1)}{e(ccp_2, cp_2)}$, and $Tr = Tr_j \cdot cp_1$. It then verifies the data index by checking $CINDX_i = \frac{e(Tr, cp_1)}{e(ccp_2, ccp_3)}$. Finally, FN_i encrypts $\{cp_1, ccp_3\}$ using the session key $sk_{FN_i/j}$ and forwards it to FN_j .
- S6: FN_j decrypts $\{cp_1, ccp_3\}$ using $sk_{FN_j/i}$, and re-encrypts them with the session key sk_{FN_j-DU} , and transmit the result to the DU . Upon decryption, the DU recovers the data D as $ccp_3 \cdot e(akey, cp_1)$.

Correctness of Data Decryption

Proof.

$$\begin{aligned}
 & p_3 \cdot e(akey, cp_1) \\
 &= cp_3 \cdot \frac{e(ccp_1, cp_1)}{e(ccp_2, cp_2)} \cdot e(akey, cp_1) \\
 &= cp_3 \frac{e(akey \cdot ccp_1, cp_1)}{e(ccp_2, cp_2)} \\
 &= \frac{cp_3 \cdot e(akey \cdot \prod_{k \in S, k \neq i} g_{n+1-k+i}, g^{r_1})}{e(\prod_{k \in D} g_{n+1-k}, (g_i \cdot pk_{SDO})^{r_1})} \\
 &= \frac{cp_3 \cdot e(akey, g^{r_1}) \cdot e(\prod_{k \in D, k \neq i} g_{n+1-k+i}, g^{r_1})}{e(\prod_{k \in D} g_{n+1-k}, g^{r_{SDO} \cdot r_1}) \cdot e(\prod_{k \in D} g_{n+1-k}, g_i^{r_1})} \\
 &= \frac{cp_3 \cdot e(akey, g^{r_1})}{e(\prod_{k \in D} g_{n+1-k}, g^{r_{SDO} \cdot r_1}) \cdot e(g_{n+1}, g^{r_1})} \\
 &= \frac{Data \cdot e(g_1, g_n)^{r_1} \cdot e(\prod_{k \in S} g_{n+1-k}^{r_{SDO}}, g^{r_1})}{e(\prod_{k \in D} g_{n+1-k}, g^{r_{SDO} \cdot r_1}) \cdot e(g_1, g_n^{r_1})} \\
 &= Data
 \end{aligned}$$

□

- S1: We begin with the expression $p_3 \cdot e(akey, cp_1)$ and substitute the definition of p_3 . The expression then becomes

$$\begin{aligned}
 & p_3 \cdot e(akey, cp_1) \\
 &= cp_3 \cdot \frac{e(ccp_1, cp_1)}{e(ccp_2, cp_2)} \cdot e(akey, cp_1).
 \end{aligned}$$

- S2: Using the bilinearity property of the pairing, specifically $e(XY, Z) = e(X, Z)e(Y, Z)$, we combine the two pairings involving cp_1 as follows:

$$= cp_3 \cdot \frac{e(akey \cdot ccp_1, cp_1)}{e(ccp_2, cp_2)}.$$

This transformation merges structurally related factors and prepares the expression for substitution of the explicit form of ccp_1 .

- S3: We expand the internal product $akey \cdot ccp_1$ according to the ciphertext definition:

$$= cp_3 \cdot \frac{e(akey \cdot \prod_{k \in S, k \neq i} g_{n+1-k+i} \cdot g_1^{r_1}, cp_1)}{e(\prod_{k \in D} g_{n+1-k} \cdot (g_i \cdot pk_{SDO})^{r_1}, cp_2)}.$$

This step makes explicit the dependency of the expression on the structural index sets associated with the ciphertext components.

- S4: Exploiting the multiplicative distributivity of the pairing, $e(AB, C) = e(A, C)e(B, C)$, we distribute the pairing over all factors in both numerator and denominator:

$$= \frac{cp_3 \cdot e(akey, g_1^{r_1}) \cdot e(\prod_{k \in D, k \neq i} g_{n+1-k+i}, g_1^{r_1})}{e(\prod_{k \in D} g_{n+1-k}, g^{r_{SDO} r_1}) \cdot e(\prod_{k \in D} g_{n+1-k}, g_i^{r_1})}.$$

- S5: Rearranging symmetric terms and isolating cancellable components, we obtain

$$= \frac{cp_3 \cdot e(akey, g_1^{r_1})}{e(\prod_{k \in D} g_{n+1-k}, g^{r_{SDO} r_1})} \cdot e(g_{n+1}, g_1^{r_1}),$$

where duplicated factors in the distributed pairings have been consolidated.

- S6: We substitute the definition of the ciphertext component $cp_3 = Data \cdot e(g_1, g_n)^{r_1}$, obtaining

$$= Data \cdot e(g_1, g_n)^{r_1} \cdot \frac{e(\prod_{k \in S} g^{r_{SDO}}, g_1^{r_1})}{e(\prod_{k \in D} g_{n+1-k}, g^{r_{SDO} r_1})}.$$

- S7: Finally, we complete the derivation by canceling the symmetric pairing terms. The bilinear map satisfies $e(g, h^x) = e(g, h)^x$. This rule implies $e(g_1, g_n)^{r_1} = e(g_1, g_n^{r_1})$. Both terms appear in opposite positions in the expression. As a result, they cancel. The entire expression then reduces to $Data$.

5. Security Analysis

In this section, we evaluate the security strength of the proposed protocol through multiple analyses, including informal reasoning, the ROR model [16], and formal verification using the AVISPA tool [17].

5.1. ROR Model

In most authentication protocols, each entity verifies the legitimacy of its communication partner and establishes a shared session key. To assess the security of this session key, the ROR model [16] is employed. This model enables the evaluation of a protocol's resistance against both passive and active adversarial behaviors. Specifically, the adversary interacts with instantiated participants through a series of games and attempts to distinguish between a random nonce and a real session key using a test query.

To formally analyze session key security under the ROR model, we define the participating entities, the adversary $\mathcal{A}$, and the corresponding query operations. In particular, we focus our formal security analysis on the key agreement and aggregate key sharing processes between the DU and DO . This analysis involves two entities: data owner P_{DO}^i

and data user P_{DU}^i , where i represent specific instance of each participants. The adversary $\mathcal{A}$ is assumed to have full control over public communication channels, with the capability to intercept, modify, delete, and replay transmitted messages. Based on these capabilities, $\mathcal{A}$ can perform the following types of queries:

- $Execute(\Pi_{DO}^{t_1}, \Pi_{DU}^{t_2})$: This query models a passive attack, in which the adversary $\mathcal{A}$ eavesdrops on the message exchanges between DO and DU over an insecure public channel. The $\mathcal{A}$ can record all transmitted data without altering any message content.
- $Send(P^t, msg)$: This query represents an active attack, allowing $\mathcal{A}$ to inject crafted messages into an oracle session P^t and observe the resulting responses. This enables $\mathcal{A}$ to attempt to manipulate or replay protocol messages.
- $S(PM^a)$: This query initiates the security experiment. A fair coin $c \in \{0, 1\}$ is flipped to determine the response. If the session key ssk shared between P^t and $\mathcal{A}$ is considered fresh, then $\mathcal{A}$ receives a random value when $c = 0$, and the actual ssk when $c = 1$. If the freshness condition is not met, the oracle returns $\perp$.

ROR Proof

In accordance with the ECDDHP and ECDLP previously defined in Section Elliptic Curve Cryptography, this game is executed to prove the Theorem 1 presented below and to confirm the session key security of the proposed scheme.

Theorem 1. Assume that $\mathcal{A}$ is an adversary operating within polynomial time t against our scheme P . Then, $\mathcal{A}$'s advantage in violating the semantic security of P is expressed as:

$$Adv_P^{ECDDHP} \leq \frac{q_h^2}{|Hash|} + 2Adv^{ECDDHP}(t)$$

Here, q_h and q_s correspond to the number of hash and send queries issued by the adversary, respectively. The function $Hash$ denotes the range of a collision-resistant hash function H , while $Adv^{ECDDHP}(t)$ represents the success probability of $\mathcal{A}$ in solving the ECDDHP within polynomial time t .

Proof. The proof proceeds through a sequence of four games, G_i ($i = 0, 1, 2, 3$), where $WinA_i$ denotes the event that the adversary $\mathcal{A}$ wins the corresponding game. The formal reasoning for each game is detailed below.

- Game G_0 : The initial game, denoted as G_0 , models the real interaction between the adversary $\mathcal{A}$ and the our protocol. A random bit c is chosen at the beginning of the game, and the corresponding winning advantage of $\mathcal{A}$ is computed as:

$$Adv_P^{ECDDHP} = |2 \cdot Pr[WinA_0] - 1| \quad (1)$$

- Game G_1 : In this game, denoted G_1 , the adversary $\mathcal{A}$ passively eavesdrops on all communications within our scheme using the $Execute(P_{DO}^{t_1}, P_{DU}^{t_2})$ query. It then invokes $Test(\Pi^t)$ to decide whether the returned value represents the true session key ssk or a random bitstring. The session key ssk , computed as $sk_{DO-DU} = rn_2 \cdot CT_1^{DU}$ and $sk_{DU-DO} = rn_1 \cdot CT_1^{DO}$, can only be derived by obtaining the secret keys of both DO and DU , which is infeasible under the ECDDHP assumption. Therefore, $\mathcal{A}$'s advantage in this game satisfies the bound:

$$Pr[WinA_1] = Pr[WinA_0] \quad (2)$$

- Game G_2 : Game G_2 represents an active attack in which the adversary $\mathcal{A}$ interacts with the system by issuing $Send(P^t, msg)$ and $Hash$ queries. In this phase, $\mathcal{A}$ attempts to impersonate a legitimate participant (DO or DU) by manipulating multiple $Hash$

queries. However, $\mathcal{A}$ is unable to derive the participants' secret keys or the random value rn , since both rely on the hardness of the ECDDHP and ECDLP problems. Moreover, due to the collision resistance of the hash function, $\mathcal{A}$ cannot produce a valid hash collision within polynomial time. Consequently, by applying the birthday paradox, the adversary's advantage can be expressed as follows:

$$|Pr[WinA_1] - Pr[WinA_2]| \leq \frac{q_h^2}{2|Hash|} \quad (3)$$

- Game G_3 : The final game, denoted G_3 , simulates an extended active attack scenario. Similarly to the conditions in Game G_1 , $\mathcal{A}$ must recover both sk_{DO} and sk_{DU} to reconstruct the correct session key ssk . Nevertheless, even after executing the eavesdropping attack, distinguishing these keys from $sk_{DO-DU} = rn_2 \cdot CT_1^{DU}$ or $sk_{DU-DO} = rn_1 \cdot CT_1^{DO}$, would require solving the ECDDHP. As this problem remains computationally infeasible, $\mathcal{A}$ cannot obtain the valid session key, implying that

$$|Pr[WinA_2] - Pr[WinA_3]| \leq Adv_P^{ECDDHP}(t) \quad (4)$$

Once all four games (G_0, G_1, G_2, G_3) have been executed, $\mathcal{A}$'s final task is to determine the correct value of the bit c . Given this setting, the advantage of $\mathcal{A}$ in distinguishing the real session key from a random one is given by

$$Adv_{P,G_3}^{ECDDHP} = \frac{1}{2} \quad (5)$$

From Equations (1) and (2), the following result can be derived.

$$\frac{1}{2} \cdot Adv_P^{ECDDHP} = |Pr[WinA_0] - \frac{1}{2}| = |Pr[WinA_1] - \frac{1}{2}| \quad (6)$$

Using the triangular inequality along with Equations (3)–(5), the following conclusion can be drawn:

$$\begin{aligned} |Pr[WinA_1] - \frac{1}{2}| &= |Pr[WinA_1] - Pr[WinA_3]| \\ &\leq |Pr[WinA_1] - Pr[WinA_2]| \\ &\quad + |Pr[WinA_2] - Pr[WinA_3]| \\ &\leq \frac{q_h^2}{2|Hash|} + Adv_P^{ECDDHP}(t) \end{aligned} \quad (7)$$

Subsequently, multiplying both sides of Equation (7) by 2 provides the final result.

$$Adv_P^{ECDDHP} \leq \frac{q_h^2}{|Hash|} + 2Adv^{ECDDHP}(t)$$

□

5.2. Informal Analysis

5.2.1. Impersonation Attacks

In an impersonation attack, an adversary $\mathcal{A}$ attempts to masquerade as a legitimate DO or DU to gain unauthorized access to the DO's data. However, such an attack is infeasible in our scheme, as the adversary $\mathcal{A}$ is unable to derive the shared key $sk_{DO-DU} = rn_2 \cdot CT_1^{DU}$, which relies on secret values accessible only to legitimate participants. Obtaining a valid secret key would require solving the ECDDHP, while deriving valid authentication parameters would involve computing the proof values $rn_1, rn_2, CT_2^{DO} = rn_2 + h(m_1) \cdot sk_{DO}$.

As these depend on private random values, such computations are infeasible, ensuring robustness against impersonation attacks.

5.2.2. Replay Attacks

A replay attack occurs when an adversary intercepts and re-transmits valid messages to gain data access. Our scheme prevents such attacks by including random number in important verification messages to verify freshness. Even if $\mathcal{A}$ attempts to reuse previous messages, it would need to derive the shared key sk_{DO-DU} , the random number $\{rn_1, rn_2\}$, and the secret key $\{sk_{DO}, sk_{DU}\}$, which is infeasible. Therefore, the proposed scheme remains secure against replay attacks.

5.2.3. Man-in-the-Middle Attacks

Section 5.2.1 shows that the adversary $\mathcal{A}$ cannot derive the shared key $sk_{DO-DU} = rn_2 \cdot CT_1^{DU}$ or the proof parameters $CT_1^{DO} = rn_1 \cdot G, rn_1, rn_2, CT_2^{DO} = rn_2 + h(m_1) \cdot sk_{DO}$ without solving the ECDDHP. As a result, $\mathcal{A}$ is unable to reconstruct valid verification parameters or impersonate legitimate entities. Consequently, the proposed scheme is resistant to man-in-the-middle attacks.

5.2.4. Privileged Insider Attacks

This attack arises when a FN associated with a TTP attempts to compromise the confidentiality of data belonging to the DO or DU . In the proposed scheme, no centralized TTP is involved in managing the entities' key pairs, owing to the DIDs. Each participant independently generates, maintains, and controls its own private credentials. Consequently, the absence of centralized key management effectively eliminates insider privileges, making the scheme inherently resilient against privileged insider attacks.

5.2.5. Fine-Grained Control and Self-Sovereignty

In the proposed scheme, the DO and DU initialize system parameters independently of any TTP. Data is secured using KASE, with ciphertexts maintained in the FN . The KASE mechanism enables flexible ciphertext updates and single-key decryption across multiple datasets. Because users locally create their own key pairs, the design eliminates reliance on a key generation center or central trust body, ensuring decentralized, fine-grained, and self-sovereign access management.

5.2.6. Providing Password Update Mechanism

In the proposed scheme, entities do not need to request assistance from a TTP to update their secret keys. Each participant can independently renew its key pair and corresponding public parameters by updating the associated public/private key entries within its own DID document on the blockchain.

5.2.7. Secure Mutual Authentication

Sections 5.2.1–5.2.3 demonstrate that deriving secret parameters is computationally infeasible for an adversary $\mathcal{A}$ without solving the ECDDHP. In the data request and aggregate key sharing process, the DO and DU derive session keys $sk_{DO-DU} = rn_2 \cdot CT_1^{DU}$ and $sk_{DU-DO} = rn_1 \cdot CT_1^{DO}$ based on the intractability of the ECDLP and ECDDHP. Consequently, the proposed scheme guarantees secure mutual authentication.

5.3. Formal Security Verification Using AVISPA

In this section, the proposed scheme is implemented and verified using the AVISPA tool [17], an AVISPA-based formal analysis extensively adopted in cryptographic protocol research. AVISPA evaluates the robustness of security schemes, focusing on their resistance

to replay and man-in-the-middle attacks. Protocols are first described using the High-Level Protocol Specification Language (HPSL) [37] and automatically converted into an Intermediate Format (IF) by the HPSL2IF translator. The generated IF model is analyzed through one of four verification engines—OFMC [38], CL-AtSe [39], SATMC [40], or TA4SP [41]. Each backend provides an independent simulation report verifying the security soundness of the proposed protocol.

Simulation Results

As shown in Figures 3 and 4, the OFMC and CL-AtSe backends report consistent verification results. The analysis confirms that the proposed *DO-DU* aggregate key sharing process withstands potential replay and man-in-the-middle attacks within the specified threat models [42].

```

1 SUMMARY
2   SAFE
3 DETAILS
4   BOUNDED_NUMBER_OF_SESSIONS
5 PROTOCOL
6   /home/span/span/testsuite/results↔
   /consumer_our.if
7
8 GOAL
9   as_specified
10
11 BACKEND
12   OFMC
13
14 COMMENTS
15
16 STATISTICS
17
18   parseTime: 0.00s
19   searchTime: 22.31s
20   visitedNodes: 1091 nodes
21   depth: 9 plies

```

Figure 3. Result of OFMC.

```

1 SUMMARY
2   SAFE
3
4 DETAILS
5   BOUNDED_NUMBER_OF_SESSIONS
6   TYPED_MODEL
7
8 PROTOCOL
9   /home/span/span/testsuite/results↔
   /consumer_our.if
10
11 GOAL
12   As Specified
13
14 BACKEND
15   CL-AtSe
16
17 STATISTICS
18
19   Analysed   : 0 states
20   Reachable  : 0 states
21   Translation: 0.05 seconds
22   Computation: 0.02 seconds

```

Figure 4. Result of CLAtSe.

6. Comparative Analysis

In this section, we compare the performance and security features of our scheme with those of recent schemes in [10,31–34].

Limitations: This study aims to design a theoretically sound security protocol applicable to real-world industrial environments. However, implementing and testing the proposed scheme in a full-scale system equipped with an actual fog computing infrastructure remains challenging due to practical constraints such as resource availability and deployment cost. Therefore, the experimental evaluation was conducted in a controlled testbed environment using a Raspberry Pi platform, which serves as a lightweight representative model for mobile or edge devices. While this setup effectively validates the feasibility and performance of the proposed scheme, further large-scale deployment and real-world verification will be pursued in future work.

6.1. Security Features

Table 3 presents a summary of the security features of the proposed scheme alongside existing schemes [10,31–34]. Earlier studies overlook several vital security aspects; however, the proposed scheme successfully meets all major security objectives while maintaining decentralization and supporting fine-grained access control without the need for a TTP.

Table 3. Security features comparison.

Security Functionalities	Guo et al. [32]	Vangala et al. [33]	Ponnuru et al. [34]	Niu et al. [10]	Trivedi and Patel [31]	Ours
SF_1	○	○	○	○	○	○
SF_2	○	○	○	○	○	○
SF_3	○	○	○	○	○	○
SF_4	○	○	○	○	○	○
SF_5	○	○	○	○	○	○
SF_6	×	×	×	○	○	○
SF_7	×	×	×	×	×	○
SF_8	×	×	×	○	○	○
SF_9	○	○	○	○	○	○
SF_{10}	○	○	○	×	×	○

○: guarantee the security features, ×: does not guarantee the security features, SF_1 : Impersonation attack; SF_2 : Replay attack; SF_3 : Man-in-the-middle attack; SF_4 : Privileged Insider attack; SF_5 : Ensure secure mutual authentication; SF_6 : Support fine-grain access control; SF_7 : Support self-sovereignty; SF_8 : Perform formal security analysis (simulation); SF_9 : Perform formal security analysis (Mathematical); SF_{10} : Support decentralization.

6.2. Computational Cost

The performance evaluation was conducted on a Raspberry Pi 4B platform equipped with a 64-bit quad-core 1.5 GHz processor, 8 GB memory, and Ubuntu 20.04.2 LTS. The MIRACL [18] cryptographic library was employed to implement the proposed scheme. This configuration enabled direct measurement of computational overhead and facilitated comparison with previously proposed schemes.

The metrics T_h , T_{sm} , T_{sa} , T_{bp} , T_{exp} , and T_{PUF} represent the mean execution times for hashing, scalar point multiplication and addition, bilinear pairing, modular exponentiation, and PUF operations, respectively. Each cryptographic primitive was executed 100 times to obtain reliable averages. Table 4 summarizes the computational requirements, and Table 5 reports the comparative evaluation results.

Table 4. Performance evaluation—computation cost.

Cryptographic Primitive	Executed Time
T_h	0.014 ms
T_{sm}	2.302 ms
T_{sa}	0.009 ms
T_{bp}	6.51 ms
T_{exp}	0.857 ms
T_{PUF}	0.522 ms

The assessment involves three major operational phases: (1) data encryption and upload conducted by the *DO*, (2) search and ciphertext retrieval performed by the *FN* or server, and (3) decryption by the *DU*. The parameters m , $\bar{m}$, and α indicate the total number of indices generated, search results obtained, and ciphertexts decrypted, respectively.

For comparison with existing fog-assisted schemes that do not adopt KASE, the computation performed by users was treated as data encryption and upload, the fog node's intermediary tasks were regarded as data sharing, and the server's operations were considered as data decryption when estimating total computational overhead.

Table 5. Comparative Analysis—Computational Cost.

Schemes	Data Encryption and Upload	Data Sharing	Decryption
Guo et al. [32]	$14T_h + 2T_{sm} + T_{PUF}$ ≈ 5.322 ms	N/A	$13T_h + T_{sm}$ ≈ 2.484 ms
Vangala et al. [33]	$7T_h + 4T_{sm} + T_{sa}$ ≈ 9.315 ms	$19T_h + 9T_{sm} + 2T_{sa}$ ≈ 21.002 ms	$10T_h + 5T_{sm} + 2T_{sa}$ ≈ 11.668 ms
Ponnuru et al. [34]	$11T_h + 4T_{sm}$ ≈ 9.362 ms	$8T_h$ ≈ 0.112 ms	$10T_h + T_{sm}$ ≈ 2.442 ms
Niu et al. [10]	$n(2T_{bp} + 3T_{exp} + 6T_{sm} + 2T_h) + 2T_{exp} + T_{sa}$ $\approx n(29.417) + 1.723$ ms	$\tilde{m}(mT_{sm} + 4T_{bp} + 2T_{exp} + 4T_{sm})$ $\approx \tilde{m}(m2.302 + 36.962)$ ms	$\alpha(2T_{bp} + 4T_{sm} + T_{sa} + mT_{sm} + mT_{sa})$ $\approx 22.237 + m2.302 + m0.009$ ms
Trivedi and Patel [31]	$n(2T_{bp} + 4T_{exp} + 2T_{sm} + T_h)$ $\approx n(17.638)$ ms	$\tilde{m}(mT_{sm} + 2T_{bp} + 3T_{sm})$ $\approx \tilde{m}(m2.302 + 19.926)$ ms	$\alpha(2T_{bp} + 3T_{sm})$ $\approx \alpha(19.926)$ ms
Ours	$n(ind (T_h + T_{bp})) + T3_{exp}$ $\approx n(ind(6.524)) + 2.571$ ms	$\tilde{m}(m2T_{exp}) + 2T_h + 6T_{sm} + 2T_{sa}$ $\approx \tilde{m}(m1.714) + 13.858$	$3T_{bp} + T_{sm}$ ≈ 21.832

While the proposed scheme incurs slightly higher computational cost than prior schemes, it achieves broader and more robust security guarantees. Schemes such as those by Guo et al. [32], Vangala et al. [33], and Ponnuru et al. [34] lack fine-grained access control and formal analysis, whereas Niu et al. [10] and Trivedi and Patel [31] do not support decentralization. Moreover, none of the previous schemes preserve data self-sovereignty. In KASE-based approaches, the computational cost varies depending on the number of keywords associated with the requested data; as the number of keywords increases, the computation overhead also grows. However, schemes that do not adopt KASE cannot achieve the same level of fine-grained and flexible data access control. Thus, our scheme attains comprehensive security while ensuring decentralized operation and self-sovereign data management.

7. Conclusions

The proposed integration allows data owners, data users, and fog nodes to independently manage their identities and cryptographic credentials without reliance on a TTP. Security verification through informal analysis, the Real-or-Random model, and the AVISPA tool confirmed resistance to major network attacks, including replay, impersonation, and man-in-the-middle scenarios. Moreover, implementation results using the MIRACL library demonstrate that the proposed scheme achieves strong security with only moderate computational overhead compared with existing methods. Overall, the proposed scheme achieves secure authentication, fine-grained access control, and data self-sovereignty for industrial IoT networks. The proposed scheme provides a promising foundation for building scalable and trustworthy IIoT infrastructures in next-generation industrial ecosystems.

Funding: This research received no external funding.

Data Availability Statement: No new data were created or analyzed in this study.

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

References

1. Botta, A.; De Donato, W.; Persico, V.; Pescapé, A. Integration of cloud computing and Internet of Things: A survey. *Future Gener. Comput. Syst.* **2016**, *56*, 684–700. [\[CrossRef\]](#)
2. Hu, P.; Dhelim, S.; Ning, H.; Qiu, T. Survey on fog computing: Architecture, key technologies, applications and open issues. *J. Netw. Comput. Appl.* **2017**, *98*, 27–42. [\[CrossRef\]](#)
3. Song, D.X.; Wagner, D.; Perrig, A. Practical techniques for searches on encrypted data. In Proceedings of the IEEE Symposium on Security and Privacy 2025, Berkeley, CA, USA, 14–17 May 2020; IEEE Computer Society: Washington, DC, USA, 2000; pp. 44–55.
4. Boneh, D.; Di Crescenzo, G.; Ostrovsky, R.; Persiano, G. Public key encryption with keyword search. In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, 2–6 May 2004; Springer: Berlin/Heidelberg, Germany, 2004; pp. 506–522.
5. Li, J.; Lin, X.; Zhang, Y.; Han, J. KSF-OABE: Outsourced attribute-based encryption with keyword search function for cloud storage. *IEEE Trans. Serv. Comput.* **2017**, *10*, 715–725. [\[CrossRef\]](#)
6. Hoang, T.; Yavuz, A.A.; Guajardo, J. A secure searchable encryption framework for privacy-critical cloud storage services. *IEEE Trans. Serv. Comput.* **2021**, *14*, 1675–1689. [\[CrossRef\]](#)
7. Yin, H.; Zhang, W.; Deng, H.; Qin, Z.; Li, K. An attribute-based searchable encryption scheme for cloud-assisted IIoT. *IEEE Internet Things J.* **2023**, *10*, 11014–11023. [\[CrossRef\]](#)
8. Gao, H.; Huang, H.; Xue, L.; Xiao, F.; Li, Q. Blockchain-enabled finegrained searchable encryption with cloud-edge computing for electronic health records sharing. *IEEE Internet Things J.* **2023**, *10*, 18414–18425. [\[CrossRef\]](#)
9. Cui, B.; Liu, Z.; Wang, L. Key-aggregate searchable encryption for group data sharing via cloud storage. *IEEE Trans. Comput.* **2016**, *65*, 2374–2385. [\[CrossRef\]](#)
10. Niu, J.; Li, X.; Gao, J.; Han, Y. Blockchain-based anti-key-leakage key aggregation searchable encryption for IoT. *IEEE Internet Things J.* **2020**, *7*, 1502–1518. [\[CrossRef\]](#)
11. Liu, Z.; Li, T.; Li, P.; Jia, C.; Li, J. Verifiable searchable encryption with aggregate keys for data sharing system. *Future Gener. Comput. Syst.* **2018**, *78*, 778–788. [\[CrossRef\]](#)
12. Rajasekaran, A.S.; Azees, M.; Al-Turjman, F. A comprehensive survey on blockchain technology. *Sustain. Energy Technol. Assess.* **2022**, *52*, 102039. [\[CrossRef\]](#)
13. Babu, P.R.; Palaniswamy, B.; Reddy, A.G.; Odelu, V.; Kim, H.S. A survey on security challenges and protocols of electric vehicle dynamic charging system. *Secur. Privacy* **2022**, *5*, e210. [\[CrossRef\]](#)
14. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-Policy Attribute-Based Encryption. In Proceedings of the IEEE Symposium on Security and Privacy (SP'07), Berkeley, CA, USA, 20–23 May 2007; IEEE Computer Society: Washington, DC, USA, 2007; pp. 321–334.
15. Sporny, M.; Longley, D.; Sabadello, M.; Reed, D.; Steele, O.; Allen, C. Decentralized Identifiers (DIDs) v1.1, W3C Proposed Recommendation, July 2025. Available online: <https://www.w3.org/TR/did-1.1/> (accessed on 3 September 2025).
16. Abdalla, M.; Fouque, P.; Pointcheval, D. Password-based authenticated key exchange in the three-party setting. In *Public Key Cryptography—PKC 2005, Proceedings of the 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, 23–26 January 2005*; Lecture Notes in Computer Science (LNCS); Springer: Berlin/Heidelberg, Germany, 2005; pp. 65–84.
17. Automated Validation of Internet Security Protocols and Applications. Available online: <https://people.irisa.fr/Thomas.Genet/span/> (accessed on 3 September 2025).
18. MIRACL Cryptographic SDK. Available online: <https://github.com/miracl/MIRACL> (accessed on 19 August 2025).
19. Shuai, M.; Xiong, L.; Wang, C.; Yu, N. A secure authentication scheme with forward secrecy for Industrial Internet of Things using Rabin cryptosystem. *Comput. Commun.* **2020**, *160*, 215–227. [\[CrossRef\]](#)
20. Srinivas, J.; Das, A.K.; Wazid, M.; Kumar, N. Anonymous lightweight chaotic map-based authenticated key agreement protocol for Industrial Internet of Things. *IEEE Trans. Depend. Secure Comput.* **2018**, *17*, 1133–1146. [\[CrossRef\]](#)
21. Wazid, M.; Das, A.K.; Odelu, V.; Kumar, N.; Susilo, W. Secure remote user authenticated key establishment protocol for smart home environment. *IEEE Trans. Depend. Secure Comput.* **2017**, *17*, 391–406. [\[CrossRef\]](#)
22. Banerjee, S.; Odelu, V.; Das, A.K.; Chattopadhyay, S.; Giri, D. Anonymous Fine-Grained User Access Control Scheme for Internet of Things Architecture. In Proceedings of the 5th International Conference on Mathematics and Computing, Bhubaneswar, India, 6–9 February 2019; pp. 1–18.
23. Mubarakali, A. Healthcare services monitoring in cloud using secure and robust healthcare-based BLOCKCHAIN (SRHB) approach. *Mobile Netw. Appl.* **2020**, *25*, 1330–1337. [\[CrossRef\]](#)
24. Park, K.; Park, Y. MIIoT-CDPS: Complete decentralized privacy-preserving scheme for medical internet of things. *Internet Things* **2024**, *27*, 101250. [\[CrossRef\]](#)
25. Dolev, D.; Yao, A.C. On the security of public key protocols. *IEEE Trans. Inf. Theory* **1983**, *29*, 198–208. [\[CrossRef\]](#)

26. Eddine, M.S.; Ferrag, M.A.; Friha, O.; Maglaras, L. EASBF: An efficient authentication scheme over blockchain for fog computing enabled Internet of Vehicles. *J. Inf. Secur. Appl.* **2021**, *59*, 102802.
27. Tomar, A.; Tripathi, S. Blockchain-assisted authentication and key agreement scheme for fog-based smart grid. *Clust. Comput.* **2022**, *25*, 451–468. [[CrossRef](#)]
28. Zhang, J.; Yang, Y.; Liu, X.; Ma, J. An efficient blockchain-based hierarchical data sharing for healthcare Internet of Things. *IEEE Trans. Ind. Inform.* **2022**, *18*, 7139–7150. [[CrossRef](#)]
29. Jia, X.; Luo, M.; Wang, H.; Shen, J.; He, D. A blockchain-assisted privacy-aware authentication scheme for Internet of Medical Things. *IEEE Internet Things J.* **2022**, *9*, 21838–21850. [[CrossRef](#)]
30. Karankar, N.; Seth, A. An IoT system for access control using blockchain and message queuing system. *EURASIP J. Inf. Secur.* **2025**, *2025*, 31. [[CrossRef](#)]
31. Trivedi, H.S.; Patel, S.J. Key-aggregate searchable encryption with multi-user authorization and keyword untraceability for distributed IoT healthcare systems. *Trans. Emerg. Telecommun. Technol.* **2023**, *34*, 4. [[CrossRef](#)]
32. Guo, Y.; Zhang, Z.; Guo, Y.; Xiong, P. BSRA: Blockchain-based secure remote authentication scheme for fog-enabled Internet of Things. *IEEE Internet Things J.* **2024**, *11*, 3348–3361. [[CrossRef](#)]
33. Vangala, A.; Das, A.K.; Mitra, A.; Das, S.K.; Park, Y. Blockchain-enabled authenticated key agreement scheme for mobile vehicles-assisted precision agricultural IoT networks. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 904–919. [[CrossRef](#)]
34. Ponnuru, R.B.; Kumar, S.A.P.; Azab, M.; Alavalapati, G.R. BAAP-FIoT: Blockchain-Assisted Authentication Protocol for Fog-Enabled Internet of Things Environment. *IEEE Internet Things J.* **2025**, *12*, 15681–15696. [[CrossRef](#)]
35. Lee, J.; Oh, J.; Kwon, D.; Kim, M.; Kim, K.; Park, Y. Blockchain-enabled key aggregate searchable encryption scheme for personal health record sharing with multidelegation. *IEEE Internet Things J.* **2024**, *11*, 17482–17494. [[CrossRef](#)]
36. Boneh, D.; Franklin, M. Identity-Based Encryption from the Weil Pairing. In Proceedings of the CRYPTO: Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 August 2001; pp. 213–229.
37. Oheimb, D.V. The High-Level Protocol Specification Language HLPsL Developed in the EU Project AVISPA. Available online: <https://www.davoh.de/cs/talks/AVISPA-HLPsL.pdf> (accessed on 2 August 2025).
38. Basin, D.; Modersheim, S.; Vigano, L. OFMC: A Symbolic Model Checker for Security Protocols. *Int. J. Inf. Secur.* **2005**, *4*, 181–208. [[CrossRef](#)]
39. Turuani, M. The CL-Atse Protocol Analyser. In Proceedings of the RTA: International Conference on Rewriting Techniques and Applications, Seattle, WA, USA, 12–14 August 2006; pp. 277–286.
40. Armando, A.; Compagna, L. SATMC: A SAT-Based Model Checker for Security Protocols. In Proceedings of the European Workshop on Logics in Artificial Intelligence, Lisbon, Portugal, 27–30 September 2004; pp. 730–733.
41. Boichut, Y.; Ham, P.C.; Kouchnarenko, O. *Automatic Verification of Security Protocols Using Approximations*; INRIA: Rocquencourt, France, 2005; Research Report inria-00070291.
42. Park, K. The AVISPA Simulation of Our Scheme. Available online: https://github.com/ISNPL/AVISPA_DADAC (accessed on 6 November 2025).

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.