Query Privacy in Data Spaces

Shuwen Liu
School of Data Science
The Chinese University of Hong Kong, Shenzhen, China
lsw00060@163.com

George C. Polyzos
School of Data Science
The Chinese University of Hong Kong, Shenzhen, China
and ExcID P.C., Athens, Greece
polyzos@acm.org

Abstract—**We design a privacy-preserving data proxy mechanism within the FIWARE Data Space framework, utilizing searchable encryption to ensure metadata confidentiality. The system is engineered to enable secure and efficient data querying, hiding the queries from the proxy and other data in the proxy from the querying agent. Recognizing the necessity of regulatory compliance, this paper integrates GDPR compliance modules into the FIWARE Data Space architecture, addressing data collection, storage, sharing, and erasure processes to enhance global applicability and regulatory adherence. In essence, we preserve metadata privacy. Experimental evaluations demonstrate the feasibility of the proposed query privacy mechanisms, focusing on metadata confidentiality and system scalability in data-intensive environments.**

I. Introduction

Data Spaces [1], [2], defined by the International Data Spaces Association (IDSA) [3] and European Data Space initiatives [4], [5], are emerging as a new form of digital platform aiming at enabling controlled, secure, and trusted data sharing to facilitate realizing the digital economy. A growing number of reports by commercial entities and governmental bodies highlight their business potential and the possible societal impact. The European Union has been supporting developments in this area for many years, but many other countries and regions are now embracing the vision and technologies, supporting research and development, including China.[1]

A data space is composed of building blocks that enable semantic interoperability of data, uniform data access methods, as well as increased data sovereignty and trust [5]. Data intermediaries or brokers play a key role in the data space architecture [3].

[1]It is being reported (25 Nov. 2024) that "China aims for more than 100 'trusted data spaces' by 2028 under national action plan" (https://www.scmp.com/news/china/politics/article/3287937/china-aims-more-100-trusted-data-spaces-2028-under-national-action-plan).

Workshop on Innovation in Metadata Privacy: Analysis and Construction Techniques (IMPACT) 2025
28 February 2025, San Diego, CA, USA
ISBN 979-8-9919276-2-8
https://dx.doi.org/10.14722/impact.2025.23070
www.ndss-symposium.org

Metadata privacy within data spaces has become a paramount concern. Metadata, including query patterns and user interactions, is vital for query processing and system management but poses privacy risks even when data is encrypted [6], [7]. Ensuring metadata privacy is critical for safeguarding user confidentiality and meeting regulatory requirements. Metadata is not message content but information about the communication [8]. The surge in the Internet of Things (IoT) devices has escalated the volume and variety of metadata, complicating its collection, analysis, and protection. Ensuring metadata privacy is essential not only for safeguarding individual privacy, but also for complying with stringent regulations like the General Data Protection Regulation (GDPR) and China's Personal Information Protection Law (PIPL). These mandate comprehensive data protection measures, including data minimization, transparency, and user control over personal information. However, industrial and financial data security and metadata privacy are perhaps even more important. Thus, robust mechanisms are urgently needed to preserve data integrity, availability and confidentiality, but also metadata confidentiality without compromising system performance.

Current approaches have made progress in addressing metadata privacy risks. For instance, de Montjoye et al. introduced openPDS [9], a personal metadata management framework that allows individuals to collect, store, and grant fine-grained access to their metadata. openPDS enhances privacy by implementing SafeAnswers, which converts high-dimensional metadata into low-dimensional responses, thereby mitigating re-identification risks and preserving sensitive information. However, openPDS is limited in high-dimensional, dynamic IoT environments that require real-time and detailed queries.

For decentralized systems, Greschbach et al. demonstrated [10] that metadata in Decentralized Online Social Networks (DOSNs) can expose sensitive user information through inference attacks, even when content is encrypted. They identified challenges such as leakage of stored object properties, access control mechanisms, and communication flows. Building on this, our work targets metadata privacy in query processing—a critical aspect of decentralized systems. Specifically, we aim to hide the content of user queries and the identities of users from intermediary nodes in data spaces, ensuring that neither query keywords nor user identities can be inferred by proxies or other potential adversaries.

Legal and regulatory requirements further complicate metadata privacy. Chiara noted that advanced encryption techniques often fail against metadata analysis, especially with encrypted traffic [11]. GDPR compliance introduces additional complexities, requiring comprehensive designs that address both technical and legal challenges. Chen et al. proposed Mohito [12], a scalable IoT metadata-hiding system using oblivious key-value storage to protect user-device interactions. Mohito is focused on specific IoT use cases.

Building on the above foundational studies, this paper focuses on the query privacy issues of intermediaries in a FIWARE Data Space (i.e., context brokers or proxy components), especially metadata privacy. Our key contributions include:

- Integration of Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs): Assigning each query a unique DID and generating a unique VC ensures fine-grained, trust-based access and authorization (and Zero Trust properties).
- Searchable Encryption (SE) and Metadata Obfuscation: Specifically, we utilize Searchable Symmetric Encryption (SSE) to enable direct processing of encrypted queries, maintaining confidentiality, while metadata obfuscation conceals auxiliary data such as IP addresses and user agents, minimizing leakage risks.
- Multi-layer Digital Watermark Verification: Embedding watermarks at each stage by clients, agents, and data providers ensures data integrity and authenticity throughout the transmission chain.
- Integration with the FIWARE Framework: We propose the "Extended Proxy Server," combining a traditional data proxy with FIWARE's Context Broker, which aligns with NGSI-LD standards, enabling privacy-preserving queries and metadata obfuscation while ensuring encrypted, unlinkable queries and full FIWARE interoperability.

The remainder of this paper is organized as follows: Section II reviews related work. Section III presents the system design. Section IV presents our evaluation in multiple dimensions. Finally, Section V provides brief conclusions and directions for future work.

II. Related Work

Achieving query privacy in data spaces involves several key elements: metadata privacy, cryptography, decentralized identity management, and regulatory compliance. This section reviews relevant literature, discussing major contributions and challenges.

Metadata, while enhancing data utility, also introduces privacy risks [9]. In federated learning, sharing metadata like feature names boosts model accuracy but compromises privacy [6]. Similarly, in secure VoIP communications, exposing metadata such as caller identities and call durations despite encryption presents a privacy concern [13]. These examples highlight the need for privacy-preserving mechanisms that protect metadata without sacrificing functionality.

Searchable encryption (SE) enables secure queries over encrypted data [14], protecting indexes and access patterns. However, it often encounters performance-security tradeoffs in large-scale scenarios, and identity privacy remains a key challenge. We adopt Searchable Symmetric Encryption (SSE) [15] for its sublinear search and reduced metadata leakage. SSE's index-based structures and dynamic updates preserve confidentiality without compromising performance.

Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs) are emerging standards for decentralized identity management, providing secure mechanisms for identity verification and authorization. Mazzocca et al. [13] survey DIDs and VCs, examining their implementations, applications, regulatory frameworks, and the challenges and future directions for their adoption beyond Self-Sovereign Identity systems. In the IoT context, Mahalle et al. [16] investigate decentralized identity management by analyzing various identification methods, evaluating DIDs and VCs in resource-constrained devices through a smart home case study, and conducting a threat analysis to ensure secure and scalable IoT deployments. While current solutions effectively manage identities, integrating DIDs and VCs into data spaces is complex, particularly in achieving seamless interoperability and scalability.

The General Data Protection Regulation (GDPR) has established stringent data privacy and protection standards, significantly shaping the design and implementation of data management systems. The study of Wachter [17] highlights GDPR's crucial role in addressing privacy and identifiability challenges within the IoT, demonstrating how its standards balance necessary identification and access control with user privacy rights. Similarly, the study of Zaeem et al. [18] indicates that GDPR has advanced user data protection, particularly by granting rights to edit and delete personal information. Despite these improvements, integrating GDPR compliance into dynamic and decentralized data environments remains challenging [10].

Data spaces facilitate interoperable data sharing, supporting a collaborative data economy [4]. While open standards are essential, robust privacy mechanisms are often underdeveloped. FIWARE, an open-source IoT platform, addresses this gap through components like the Context Broker and Data Space Connector, enabling privacy management based on NGSI-LD standards [19]. Recent enhancements, including FIWARE's IDS Certification for the Data Space Connector, reinforce GDPR compliance using identity management protocols like X.509, did:web, and Self-Sovereign Identity (SSI) [3].

By integrating FIWARE with advanced data connector standards and aligning with Data Space initiatives, our system addresses contemporary privacy and regulatory challenges, contributing a solution towards the maturing of the data economy.

III. System Design

A. System Architecture

The privacy-preserving data proxy system ensures secure data querying and effective metadata privacy management. It comprises three main components: Client, Extended Proxy Server, and Data Provider.

Client: Initiates queries by generating a unique Decentralized Identifier (DID) for each request, ensuring unlinkability to previous queries. Queries are encrypted using SSE, allowing the proxy server to process them without revealing their content. A timestamp is added as a digital watermark for traceability and integrity verification. The encrypted query, along with the DID and timestamp, is transmitted to the proxy server to mitigate tracking risks.

Extended Proxy Server: Acts as an innovative intermediary, integrating the data proxy with FIWARE's Context Broker to enhance functionality. It verifies the DID and timestamp to ensure query authenticity and integrity. By leveraging SSE, it retrieves relevant data from the encrypted Key-Value (KV) store. After query validation, it issues a Verifiable Credential (VC) to authorize subsequent client queries [20], [21], ensuring only the original DID-associated client can continue operations, preventing session hijacking and maintaining a secure, privacy-preserving user experience.

Data Provider: Stores encrypted data and processes queries forwarded by the proxy. It employs SE to search within encrypted storage, encrypts matching responses using symmetric encryption (e.g., AES), and returns them to the proxy. The proxy then validates and forwards the response to the client.

The system utilizes a Key-Value (KV) model for encrypted storage, where both the proxy server and data provider maintain encrypted key-value pairs. Keys are encrypted using SE, enabling encrypted query terms to serve as keys, while values store data encrypted with symmetric algorithms like AES. The proxy server maintains minimal metadata (e.g., data names and descriptions) to support query matching, whereas the data provider handles encrypted queries and responses [22]. This architecture ensures that the proxy server only indicates data availability, maintaining data confidentiality throughout the query process.

Figure 1 primarily highlights the query privacy mechanisms, focusing on the interaction during encrypted query processing.

B. Privacy-Preserving Query Mechanisms

This section details how our system implements privacy-preserving modules to protect queries and their metadata. We concentrate on the actual usage of SE and Metadata Obfuscation within our prototype environment. Figure 2 illustrates the implementation of SE in the system, utilizing an encrypted Key-Value (KV) list for query matching.

1) SSE-Based Queries in a Key-Value Store: To safeguard query privacy in potentially untrusted infrastructures [23], [24], we adopt a symmetric Searchable Encryption (SSE) scheme [22], [24] that encrypts both keywords and documents before storing them in a KV repository. We denote:

- $\mathcal{D} = \{d_1, d_2, \ldots, d_n\}$: The dataset of n encrypted documents, each stored as a KV pair.
- $\mathcal{K} = \{k_1, k_2, \ldots, k_m\}$: The set of m relevant keywords.
- $\text{Enc}_k(\cdot)$: A block cipher under key k.
- SSEIndex: The secure index that maps encrypted keywords $\text{Enc}_k(k_j)$ to sets of encrypted documents $\text{Enc}_k(d_i)$.

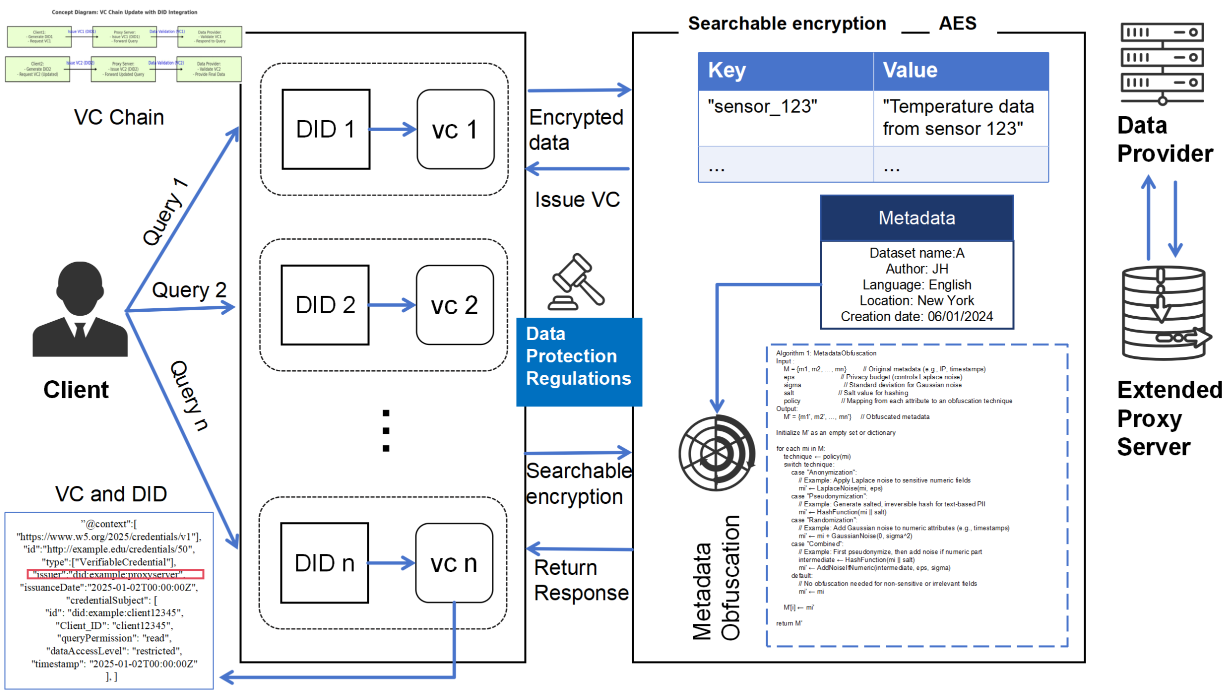
The client avoids storing plaintext KV pairs $\{k_j : d_i\}$ by computing an SSE index with pseudo-code shown in Algorithm 1. With exclusive access to the secret key k, the client encrypts each keyword k_j into tokenized entries. The server, receiving only these encrypted entries, cannot interpret the original keywords or document content. This description uses a basic "exact match" SSE scheme, which can be extended to support fuzzy or range queries [24].

Algorithm 1 Building SSEIndex in Key-Value Store

```
Require: Dataset $\mathcal{D} = \{d_1, \ldots, d_n\}$, Keyword set $\mathcal{K} = \{k_1, \ldots, k_m\}$, Symmetric key $k$
Ensure: SSEIndex = $\{\text{Enc}_k(k_j) : \{\text{Enc}_k(d_i) \ldots\}\}$
 1: SSEIndex ← $\emptyset$
 2: for each document $d_i$ in $\mathcal{D}$ do
 3:     $\text{KW}_i \leftarrow \text{ExtractKeywords}(d_i)$ {plaintext extraction}
 4:     $\text{EncDoc}_i \leftarrow \text{Enc}_k(d_i)$ {encrypt entire document}
 5:     $\text{Store}(\text{KV}, \text{EncDoc}_i)$ {put in KV store}
 6:     for each keyword kw in $\text{KW}_i$ do
 7:         $\text{TKw} \leftarrow \text{Enc}_k(\text{kw} \| r)$ {random nonce $r$ ensures distinct ciphertext}
 8:         $\text{SSEIndex}[\text{TKw}] \leftarrow \text{SSEIndex}[\text{TKw}] \cup \{\text{EncDoc}_i\}$
 9:     end for
10: end for
11: return SSEIndex
```

Here:

- Line 7 includes a random nonce r_{kw} so that $\text{Enc}_k(\text{kw}_1 \| r) \neq \text{Enc}_k(\text{kw}_1 \| r')$ [22].



Concept Diagram: VC Chain Update with DID Integration
Client1: - Generate DID1 - Request VC1
Issue VC1 (DID1)
Proxy Server: - Issue VC1 (DID1) - Forward Query
Data Validation (VC1)
Data Provider: - Validate VC1 - Respond to Query
Client2: - Generate DID2 - Request VC2 (Updated)
Issue VC2 (DID2)
Proxy Server: - Issue VC2 (DID2) - Forward Updated Query
Data Validation (VC2)
Data Provider: - Validate VC2 - Provide Final Data
VC Chain
Client
Query 1
Query 2
Query n
DID 1
vc 1
DID 2
vc 2
DID n
vc n
Encrypted data
Issue VC
Data Protection Regulations
Searchable encryption
Return Response
Searchable encryption
AES
Metadata
Metadata Obfuscation
Data Provider
Extended Proxy Server
VC and DID

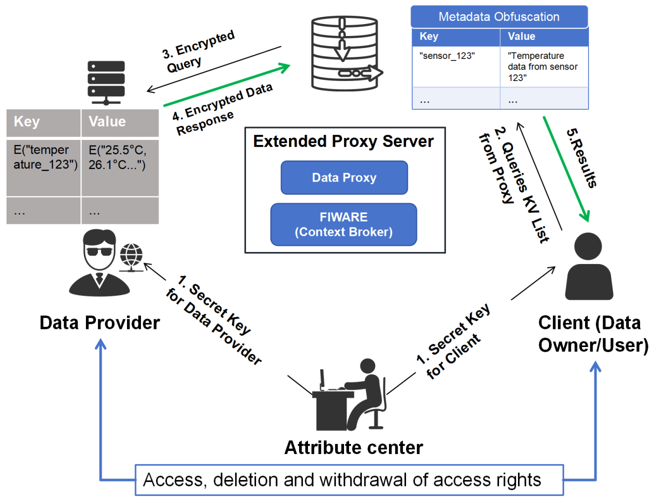
Key	Value
"sensor_123"	"Temperature data from sensor 123"
...	...

Dataset name:A
Author: JH
Language: English
Location: New York
Creation date: 06/01/2024

```
"@context":[
"https://www.w5.org/2025/credentials/v1"],
"id":"http://example.edu/credentials/50",
"type":["VerifiableCredential"],
"issuer":"did:example:proxyserver",
"issuanceDate":"2025-01-02T00:00:00Z",
"credentialSubject": [
"id": "did:example:client12345",
"Client_ID": "client12345",
"queryPermission": "read",
"dataAccessLevel": "restricted",
"timestamp": "2025-01-02T00:00:00Z"
], ]
```

```
Algorithm 1: MetadataObfuscation
Input :
   M = {m1, m2, ..., mn}        // Original metadata (e.g., IP, timestamps)
   eps                          // Privacy budget (controls Laplace noise)
   sigma                        // Standard deviation for Gaussian noise
   salt                         // Salt value for hashing
   policy                       // Mapping from each attribute to an obfuscation technique
Output:
   M' = {m1', m2', ..., mn'}    // Obfuscated metadata

Initialize M' as an empty set or dictionary

for each mi in M:
   technique ← policy(mi)
   switch technique:
      case "Anonymization":
         // Example: Apply Laplace noise to sensitive numeric fields
         mi' ← LaplaceNoise(mi, eps)
      case "Pseudonymization":
         // Example: Generate salted, irreversible hash for text-based PII
         mi' ← HashFunction(mi || salt)
      case "Randomization":
         // Example: Add Gaussian noise to numeric attributes (e.g., timestamps)
         mi' ← mi + GaussianNoise(0, sigma^2)
      case "Combined":
         // Example: First pseudonymize, then add noise if numeric part
         intermediate ← HashFunction(mi || salt)
         mi' ← AddNoiseIfNumeric(intermediate, eps, sigma)
      default:
         // No obfuscation needed for non-sensitive or irrelevant fields
         mi' ← mi

   M'[i] ← mi'

return M'
```

Fig. 1. Query system architecture overview



Key	Value
"sensor_123"	"Temperature data from sensor 123"
...	...

Key	Value
E("temper ature_123")	E("25.5°C, 26.1°C...")
...	...

Fig. 2. Searchable Encryption workflow with Key-Value store integration

- The KV store ends up holding the actual encrypted documents $\mathrm{Enc}_k(d_i)$, while the SSEIndex cross-references $\mathrm{Enc}_k(k_j)$ to $\mathrm{Enc}_k(d_i)$.

To perform a query q, the client extracts its keywords and encrypts them as $\mathrm{Enc}_k(q)$. The proxy or server matches the resulting token C_q with the SSEIndex. When a match is found, the system returns the corresponding encrypted records $\mathrm{Enc}_k(d_i)$ [23], [24]. Algorithm 2 gives a simplified representation:

Algorithm 2 SSE Querying in Our Prototype

```
Require: Query $q$, Symmetric key $k$, SSEIndex
Ensure: Set of matching ciphertexts $\{\mathrm{Enc}_k(d_i)\}$
1: tokens ← ExtractKeywords($q$)
2: for each keyword $w$ in tokens do
3:     $T_w \leftarrow \mathrm{Enc}_k(w \| r_w)$ {fresh nonce $r_w$ each time}
4:     results ← results ∪ SSEIndex[$T_w$]
5: end for
6: return results
```

In our FIWARE-based system, the Extended Proxy Server can store metadata about these SSE-protected KV entries. The proxy holds no plaintext—all matching is done over encrypted tokens T_w.

Figure 2 illustrates an SSE-based query framework integrated with a key–value repository: the attribute center issues secret keys to both data provider and client, the data provider encrypts documents into KV pairs (applying metadata obfuscation) and builds the SSE index, and the extended proxy server (combining a data proxy with FIWARE) mediates encrypted queries—ensuring only the client with the correct secret key can retrieve matching ciphertexts under strict access control.

2) Metadata Obfuscation: While SSE protects query contents, metadata (e.g., timestamps, IP addresses, user patterns) may still reveal user identities [7], [24]. Our metadata obfuscation module resides within the proxy:

- **IP Pseudonymization:** The proxy intercepts sensitive fields such as IP addresses, replacing them with pseudonyms Hash(IP $\|$ salt).
- **Timestamps:** For each request, the proxy adds a random offset $\Delta \sim N(0, \sigma^2)$ to the original timestamp to hamper correlation analyzes.
$$t' = t + \Delta. \tag{1}$$
- **Padding and Noise:** A random length of bytes is appended to each request body, making the final size less predictable. Let PadLen $\sim$ Uniform(a, b). Then the request is:
$$\text{Req} \leftarrow \text{Req} \,\|\, \text{RandomBytes(PadLen)}. \tag{2}$$

Pseudo-code for request obfuscation is as follows:

Algorithm 3 ProxyObfuscation

```
Require: Original request $R$, $(\text{salt}, \sigma, [a, b])$
Ensure: Obfuscated request $R'$
1: $(\text{IP}, t, \text{body}, \ldots) \leftarrow \text{Parse}(R)$
2: $\text{pseudoIP} \leftarrow \text{Hash}(\text{IP} \,\|\, \text{salt})$
3: $\Delta \leftarrow \text{Gaussian}(0, \sigma^2)$
4: $t' \leftarrow t + \Delta$
5: $\text{padLen} \leftarrow \text{Uniform}(a, b)$
6: $\text{body}' \leftarrow \text{body} \,\|\, \text{RandomBytes}(\text{padLen})$
7: $R' \leftarrow (\text{pseudoIP}, t', \text{body}', \text{otherMeta})$
8: return $R'$
```

Once obfuscated, the request is forwarded, so even if compromised, an adversary gains minimal insight into real user identities or query patterns. This approach aligns with the layered anonymization and differential privacy recommendations from [23], [24], [25].

3) DIDs, VCs, and FIWARE Integration: While SSE and metadata obfuscation focus on query confidentiality, we also require authorization and authenticity [13]. We incorporate DIDs and VCs to ensure that only legitimate users can submit or replay queries. The workflow is as follows:

Algorithm 4 DID+VC Flow

```
Require: DID, oldVC (or None), Timestamp
Ensure: newVC
1: if VCstore[DID] $\neq$ oldVC then
2:     reject "Invalid VC or DID"
3: else
4:     newVC $\leftarrow H(\text{DID} \,\|\, \text{oldVC} \,\|\, \text{Timestamp})$
5:     VCstore[DID] $\leftarrow$ newVC
6:     return newVC
7: end if
```

This approach links each request to a valid DID/VC chain, blocking replay and unauthorized access. The SSE scheme safeguards query content, while metadata obfuscation anonymizes sensitive fields.

C. Security Analysis

To rigorously show our system's security, we adopt an adaptive SSE model similar to [22]. This model ensures an adversary cannot recognize or decrypt encrypted queries beyond the inherent leakage profile (for instance, result sizes). Let $\mathcal{A}$ be a polynomial-time adversary that follows the honest-but-curious model. The system starts with a secret key k known only to the client. When the client queries a keyword w, it creates an ephemeral token $\mathrm{Token}(w) = \mathcal{E}_k(\, w \,\|\, \rho)$, where ρ is a fresh random nonce. The adversary may see $\mathrm{Token}(w)$, partial metadata, and returned ciphertexts, but never learns w or gains k. Under adaptive indistinguishability, if $\mathcal{A}$ provides two document sets $\{D_0, D_1\}$ and adaptively picks keywords from them, $\mathcal{A}$ cannot tell which set or which keyword is used, except with negligible advantage. This aligns with simulation-based SSE: a simulator can replicate the adversary's view using only known leakage (mostly the sizes of result sets), without revealing actual keywords.

Our design adds ephemeral DID-based credentials to stop unauthorized submissions and replay. Each query carries a DID δ and a verifiable credential ν that links δ to a time-based secret. The proxy's credential store checks (δ, ν). If valid, the store updates the credential so old ones become invalid. An attacker intercepting ν cannot reuse it, since new queries need a freshly issued credential. Even if $\mathcal{A}$ obtains a valid trapdoor $\mathrm{Token}(w)$, it cannot submit new queries without the correct ephemeral credential. Cryptographic binding of DID and ν relies on collision-resistant hashes and signature checks, so forging them is negligible in probability.

We also employ metadata obfuscation to counter linkage attacks from [26], which rely on repeated IP addresses, stable timestamps, or predictable packet lengths. For each transmission, the proxy hashes the IP with salt, pads the packet by a random amount χ, and shifts the timestamp by a Gaussian-distributed delay Δ. If $\mathcal{A}$ tries to correlate repeated queries by matching $\mathsf{IP\text{-}Timestamp\text{-}Size}$ tuples, the randomness from Δ and χ lowers correlation success below any realistic threshold (see Section IV). An attacker thus cannot reliably decide whether two obfuscated queries come from the same user or the same keyword, especially given SSE token randomization.

Each component works at a separate protocol layer with different keys, so there is no compositional conflict. The SSE secret key k is distinct from DID signing keys, and obfuscation does not reveal the trapdoor process. A breach of the DID store does not uncover SSE tokens, and manipulating the SSE index does not yield valid credentials. The combined system achieves adaptive-query confidentiality (the adversary cannot differentiate which keywords appear), prevents replay

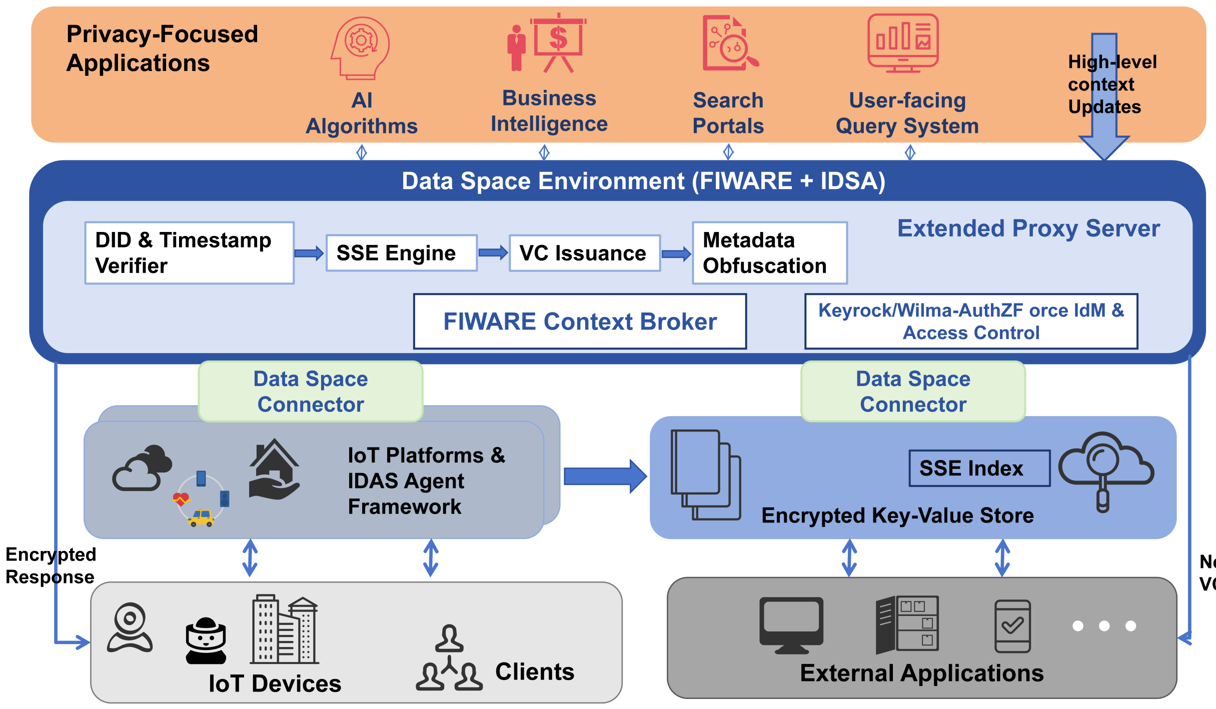


Fig. 3. Privacy-Preserving Data Space Architecture with an Extended Proxy (FIWARE + IDSA)

or stolen-credential use, and frustrates traffic-analysis based on size or timing patterns. The formal SSE framework from [22] is extended to cover ephemeral credentials and randomized metadata. Suppose $\mathcal{A}$ issues many queries $\{w_1, \ldots, w_q\}$ while observing their tokens and ephemeral credentials. If $\mathcal{A}$ could distinguish any w_i from another w_j with non-negligible probability, that would contradict SSE security plus credential constraints on query issuance. Random IP hashing and Gaussian timestamp shifts further cut the chance that repeated queries can be linked, since each request has high entropy in $\{\mathsf{IP}', \mathsf{Time}', \mathsf{Size}'\}$.

Overall, combining SSE tokens, rotating DID-based credentials, and stochastic metadata obfuscation prevents any single mechanism from leaking plaintext queries or user identities. The adaptive SSE underpinnings and ephemeral credential scheme give negligible advantage to a polynomial adversary, while the noise layer greatly reduces real-world traffic correlation. This architecture aligns with FIWARE-based data exchanges [27] and meets GDPR/PIPL demands for confidential data handling and minimized linkability.

D. *Integration with FIWARE*

Our privacy-preserving data proxy uses and extends FIWARE's open-source framework to support secure data exchange and metadata protection in Data Spaces. The FIWARE documentation [27] shows that a consistent digital-twin view needs domain-agnostic APIs and clear governance. We use the Context Broker and Data Space Connector to combine query encryption and metadata obfuscation with NGSI-LD. We now describe how the Extended Proxy Server aligns with FIWARE guidelines while keeping query confidentiality and data sovereignty.

1) Context Management via NGSI-LD: FIWARE Context Brokers (e.g., Orion-LD, Scorpio) offer an NGSI-LD interface for creating, updating, and querying context entities as digital twins. Our Extended Proxy Server adds privacy measures. Specifically, when a client sends an encrypted query with SSE, the proxy intercepts it and interacts with the Context Broker only through ciphertext-based lookups. This ensures that even if the Context Broker runs in a partially untrusted environment, it processes data without ever seeing the underlying query keywords. At the same time, the proxy applies IP pseudonymization and random timestamp shifts. NGSI-LD operations and subscriptions remain unchanged, but the broker does not see the original parameters.

2) Data Sovereignty and Trust: The FIWARE ecosystem promotes data sovereignty through Identity and Access Management (IAM) mechanisms, often combined with IDS connectors to enforce organizational-level trust. We strengthen this approach by embedding Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs) in the Extended Proxy Server. Each NGSI-LD request (GET, POST, UPDATE) must include a ephemeral DID-VC pair. The proxy verifies these credentials before decryption or forwarding. This two-tier control—(i) organizational-level authentication through the FIWARE Data Space Connector and (ii) fine-grained DID-based authorization—ensures only authorized participants can retrieve encrypted records or post new contextual data. Our solution augments FIWARE's OAuth2 or X.509 mechanisms (e.g., Keyrock) with ephemeral DID sessions, lowering replay risk and preventing misuse of valid credentials.

3) Right-Time Data Sharing and Metadata Confidentiality: FIWARE aims for real-time data exchange in dynamic scenarios (city traffic, industrial IoT, predictive maintenance). Under conventional circumstances, repeated NGSI-LD updates might reveal user behavior patterns—e.g., query frequency, IP location, or data consumption habits. We address this by adding a metadata-obfuscation layer. Concretely, each NGSI-LD request is padded to a random size and assigned a pseudonymized source identity. Since the Context Broker remains agnostic to SSE tokens and ephemeral credentials, it merely routes encrypted payloads and notifications to the correct destinations. This setup preserves normal FIWARE flows, yet it hides user identities and prevents traffic analysis based on size or frequency.

4) Extensibility with FIWARE Data Marketplace Components: Beyond direct NGSI-LD data exchange, FIWARE supports a broader "Data Marketplace" functionality, which can define terms and conditions for accessing and monetizing specific data streams. Our Extended Proxy Server is compatible with these marketplace components; once data owners publish a dataset or real-time context feed in the marketplace, the marketplace can enforce pricing or usage policies at the organizational level. Simultaneously, our proxy continues to operate at the query layer, maintaining SSE-based confidentiality and anonymizing all metadata. This allows data monetization while meeting strict data privacy regulations like GDPR and PIPL. It also ensures compliance with data-space governance standards defined in FIWARE's guidelines.

5) Alignment with European Data-Space Initiatives: Since FIWARE's principles are adopted by initiatives like GAIA-X and IDSA, our system inherits their focus on interoperability, modularity, and open standards. By embedding privacy protection directly into FIWARE components (Context Broker, Data Space Connector, IoT Agents), we reduce adoption barriers for organizations aiming for a privacy-preserving approach. The system efficiently handles sensitive queries in data-intensive settings (e.g., industrial analytics, city-scale sensor networks), aligning with the trust, transparency, and data-sovereignty goals of European Data Spaces.

Figure 3 shows our privacy-preserving architecture, where the Extended Proxy Server integrates FIWARE and IDSA components.

In summary, our privacy-preserving proxy is a drop-in extension for FIWARE-based data ecosystems. It adheres to NGSI-LD's minimal data model assumptions, maintains the original semantics of real-time data sharing, and integrates with standard FIWARE security and marketplace services. At the same time, it ensures query confidentiality and protects metadata privacy. This approach offers a practical solution for implementing secure, standards-compliant, and easily integrable privacy controls in modern Data Spaces.

IV. Evaluation

This section outlines the experimental setup, performance evaluation, and verification of protections against metadata-based threats. We compare three scenarios: direct query transmission, proxy with standard padding, and proxy with varying padding ranges to assess the privacy-overhead trade-off. Results confirm the approach's scalability and robust metadata confidentiality.

A. Experimental Setup and Feasibility

Our system consists of a client sending randomized queries, a proxy for metadata obfuscation and credential verification, and a server storing data with searchable encryption. Metadata, such as timestamps, query patterns, and IP addresses, can leak user details even with encrypted data [28]. To mitigate this, the client encrypts each query and delivers only ciphertext to a Key-Value store. When enabled, the proxy anonymizes IPs, randomizes timestamps, and applies configurable padding to hamper size-based inferences, ensuring interoperability within FIWARE infrastructures [19].

We design four cases:

- **Minimal Defense (Case 1).** The server verifies credentials for 10% of requests (*vc_check_prob=0.10*). The proxy applies large padding and random timestamps to 10% of queries (*big_pad_prob=0.10*), with minimal or no padding for the remaining 90%. Clients reuse nonces in 40% of cases (*nonce_reuse_prob=0.40*), and tokens are included in 20% of queries (*token_prob=0.20*), offering limited resistance to dictionary attacks.
- **Medium Defense (Case 2).** VC checks increase to 40% (*vc_check_prob=0.40*), and 40% of queries receive substantial padding/time offsets (*big_pad_prob=0.40*). Nonce reuse drops to 30% (*nonce_reuse_prob=0.30*), while token insertion rises to 40% (*token_prob=0.40*), improving protection against dictionary attacks.
- **Enhanced Defense (Case 3).** VC checks cover 70% of requests (*vc_check_prob=0.70*), with 70% of queries applying large padding and random timestamps (*big_pad_prob=0.70*). Nonce reuse reduces to 20% (*nonce_reuse_prob=0.20*), and token insertion reaches 60% (*token_prob=0.60*), enhancing resistance to attacks.
- **High Defense (Case 4).** Nearly all requests (95%) undergo VC checks (*vc_check_prob=0.95*). The proxy

enforces large padding and timestamps for 90% of queries (*big_pad_prob=0.90*). Nonce reuse is minimal at 10% (*nonce_reuse_prob=0.10*), while token inclusion peaks at 90% (*token_prob=0.90*), significantly reducing attack success rates.

Each scenario involves 500 queries, logged with nonce-ciphertext pairs and request sizes. We systematically evaluate replay, dictionary, linkage, and inference attacks across all configurations.

In addition to these four defense-level cases, we also compare the following four system-level scenarios to systematically evaluate how enabling Secure Searchable Encryption (SSE), Decentralized Identifiers/Verifiable Credentials (DID/VC), and obfuscation affects performance and security under varying concurrency. Our goal is to isolate the contribution of each component, illustrating their cumulative impact on privacy defenses while quantifying overhead:

1) Baseline (TLS only): We begin with a minimal configuration—standard TLS with no SSE or credential checks. This setup has the fastest response times. It acts as our reference for seeing how extra security measures affect latency and protection against attacks.

2) SSE-only: We then add SSE-based storage and queries. By comparing it to the baseline, we see how SSE alone blocks dictionary and replay attacks, though it adds some overhead. This approach isolates SSE's effect before we add more defenses.

3) SSE + DID/VC: We add DID-based credential checks to some requests. This step shows how verifying users reduces replay attacks and clarifies the cost of DID checks.

4) SSE + DID/VC + Obfuscation: Finally, we integrate substantial metadata obfuscation (padding and random timing) with both SSE and DID/VC checks. This fully fortified setup hinders traffic analysis and lowers attack success rates further, but it has the highest overhead.

By enabling SSE, DID/VC, and obfuscation in stages, we see how each layer boosts confidentiality and counters advanced threats. Combined with our defense-level cases, these scenarios indicate that the extra overhead remains reasonable for many IoT and smart-city environments. This layered design delivers strong safeguards against dictionary, replay, and traffic-analysis attacks while preserving acceptable performance.

B. Performance Analysis

Table I summarizes the system's operations. Metadata transformations remain linear in the number of fields processed, while logarithmic SSE lookups maintain rapid query handling. The encryption overhead remains lightweight even when padding is significant, preserving real-time responsiveness.

TABLE I
COMPLEXITY OF OPERATIONS IN THE SYSTEM

Component	**Operation**	**Complexity**
Metadata Obfuscation	Auxiliary metadata transformation	$O(m)$
Query Matching	Searchable encryption lookup	$O(\log n)$
Encryption Overhead	Query and response encryption	$O(q + d)$

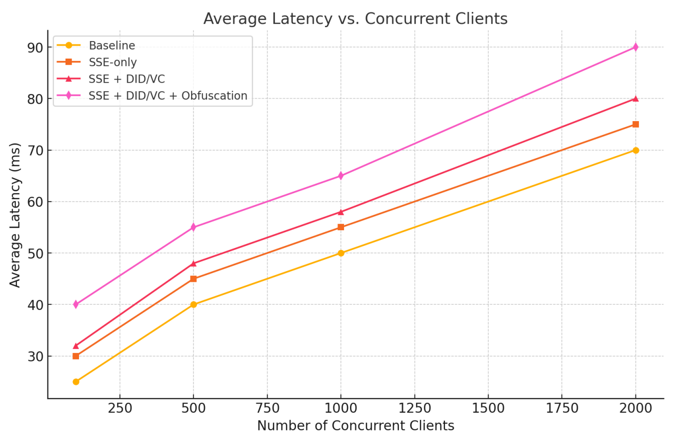


Fig. 4. Average Latency vs. Concurrent Clients

Figure 4 shows how average latency changes from 250 to 2000 concurrent clients. Under the baseline, it rises from about 25 ms at 250 clients to about 70 ms at 2000 clients. SSE-only ranges from about 30 ms to 75 ms. SSE + DID/VC goes from about 32 ms to 80 ms because of credential checks. SSE + DID/VC + Obfuscation pushes latency from about 40 ms to 90 ms because of more complex metadata transformations. Multi-layer obfuscation raises latency further, but performance stays acceptable, showing the feasibility of our approach under high concurrency.

C. Security Evaluation

Our system mitigates replay, dictionary, linkage, and inference attacks through ephemeral credential checks, SSE-based query encryption, random token insertion, and dynamic obfuscation. As shown in Figure 5, minimal defense (Case 1) results in high attack success—often exceeding 40–50%—due to sparse credential checks and limited padding, exposing exploitable patterns. Linkage and inference remain similarly elevated when timestamps or IPs remain consistent, or when nonce usage is too predictable.

Progressively increasing padding, randomizing timestamps, and embedding tokens reduces attack success rates across Cases 2–4. Replay and dictionary attacks, initially 40–60% under minimal settings, drop to 10–20% in Case 4. Linkage and inference similarly decline as greater obfuscation disrupts adversarial correlations. SSE encryption further obscures query-response pairs, mitigating the 70–80% attack rates seen without proxy measures.

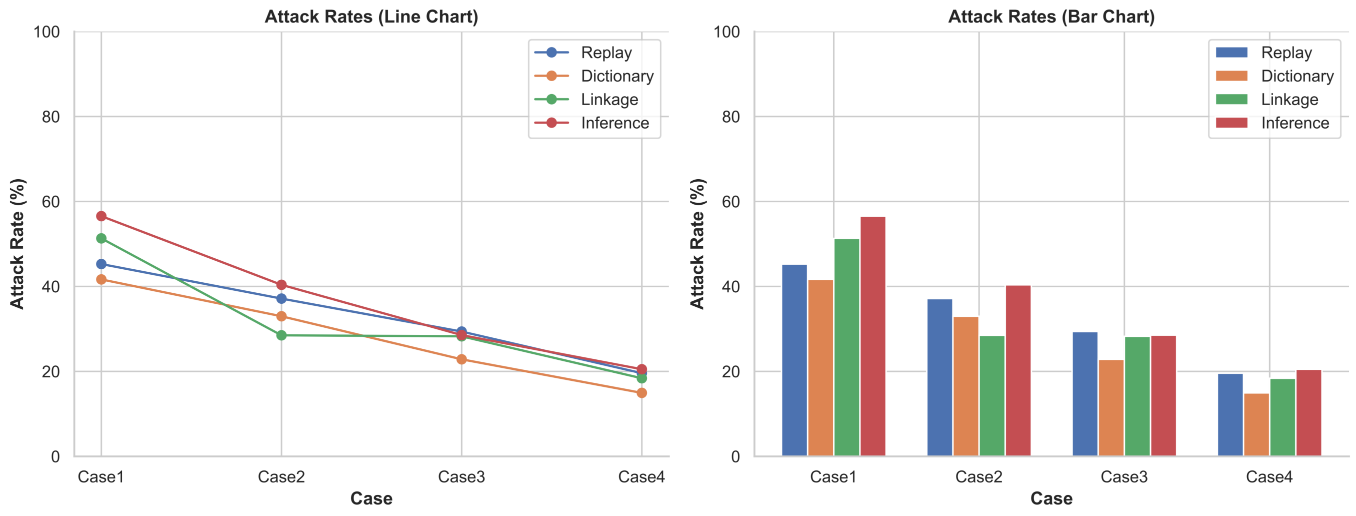


Fig. 5. Attack Success Rates Under Different Defense Scenarios

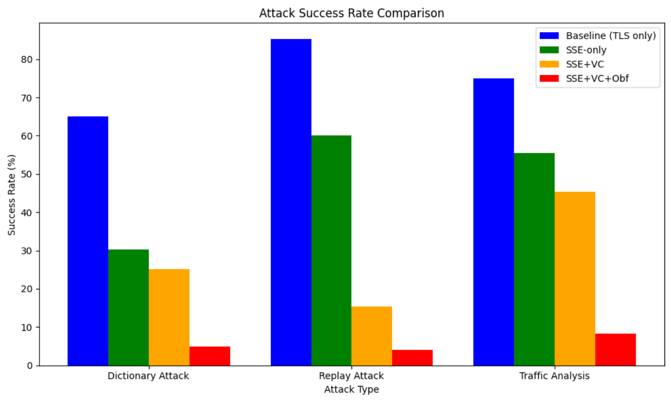


Fig. 6. Comparison of Attack Success Rates by Defense Strategy

To show security safeguards clearly, figure 6 compares four configurations—Baseline (TLS only), SSE-only, SSE + credential verification (VC), and SSE + VC + Obfuscation—tested under dictionary, replay, and traffic analysis attacks. The Baseline setup, relying solely on TLS, exhibits roughly 65% dictionary-attack success, 85% replay-attack success, and 75% traffic-analysis success. Introducing SSE encrypts both keywords and documents, substantially reducing dictionary-attack effectiveness; adding credential verification (VC) enforces stricter identity checks, mitigating replay attacks; and query obfuscation scrambles request patterns, weakening traffic analysis. As a result, SSE + VC + Obf limits these attack successes to around 5% for dictionary, 4% for replay, and 8% for traffic analysis—demonstrating the robust, layered security benefits of our proposed design.

Overall, these results show that using ephemeral credentials and obfuscation measures reduces each attack vector. Although performance latency increases with more concurrent clients, the effect is small and manageable. This ensures that the security improvements do not significantly affect efficiency. These findings highlight the effectiveness of privacy-preserving techniques in query-heavy environments, balancing security with system performance.

V. Conclusion

The designed privacy-preserving query mechanism significantly advances decentralized and interoperable Data Spaces.. By combining metadata obfuscation, searchable encryption, and decentralized identity management within the FIWARE framework, it effectively addresses critical challenges around metadata confidentiality and query privacy. Experimental results demonstrate the system's effectiveness, showing substantial reductions in the success rates of dictionary, replay, and traffic analysis attacks, even under high concurrency. These results confirm that the design delivers robust privacy protection while maintaining acceptable performance. The system's scalability, efficiency, and security make it ideal for dynamic Big Data environments, while its compliance with GDPR and PIPL further strengthens its applicability in privacy-sensitive contexts. The thorough verification of each security component confirms that the system safeguards against sophisticated attacks without compromising query efficiency. This work lays a solid foundation for building trust-focused, user-controlled data ecosystems, supporting secure, transparent, and scalable data-sharing environments. Future work will expand empirical evaluations and explore diverse data-sharing applications.

References

[1] M. Franklin, A. Halevy, and D. Maier, "From databases to dataspaces: a new abstraction for information management," *ACM SIGMOD Record*, vol. 34, no. 4, 2005. [Online]. Available: https://doi.org/10.1145/1107499.1107502

[2] E. Curry, S. Scerri, and T. Tuikka, Eds., *Data Spaces: Design, Deployment and Future Directions*. Springer Cham, 2022. [Online]. Available: https://link.springer.com/book/10.1007/978-3-030-98636-0

[3] International Data Spaces Association, "Data Connector Report," September 2024. [Online]. Available: https://internationaldataspaces.org/wp-content/uploads/dlm_uploads/IDSA-Data-Connector-Report-84-No-16-September-2024-1.pdf

[4] G. Solmaz, F. Cirillo, J. Fürst, T. Jacobs, M. Bauer, E. Kovacs, and L. Sánchez, "Enabling data spaces: Existing developments and challenges," in *Proc. 1st International Workshop on Data Economy*, 2022.

[5] "Data Spaces Blueprint v1.5," European Data Spaces Support Center, Tech. Rep., October 2024. [Online]. Available: https://dssc.eu/space/bv15e/766061169

[6] D. Zhan and R. Hai, "Will sharing metadata leak privacy?" in *Proc. IEEE Int. Conf. Data Eng. Workshops (ICDEW)*, 2024.

[7] A. Pujol, D. Magoni, L. Murphy, and C. Thorpe, "Spying on instant messaging servers: Potential privacy leaks through metadata," *Transactions on Data Privacy*, vol. 12, pp. 175–206, 2019.

[8] M. C. Domenech, A. R. A. Grégio, and L. C. E. D. Bona, "On metadata privacy in instant messaging," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, 2022.

[9] Y. A. de Montjoye, E. Shmueli, S. S. Wang, and A. S. Pentland, "OpenPDS: Protecting the privacy of metadata through safeanswers," *PLOS One*, vol. 9, no. 7, 2014.

[10] S. Buchegger, B. Greschbach, and G. Kreitz, "The devil is in the metadata—new privacy challenges in Decentralized Online Social Networks," in *Proc. IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2012.

[11] P. G. Chiara, "Privacy and data protection challenges in iot data and metadata processing," in *Internet of Things and EU Law: Cybersecurity, Privacy and Data Protection Challenges*, 2024.

[12] Y. Chen, D. Heath, R. Chatterjee, and E. Fernandes, "Scalable metadata-hiding for privacy-preserving iot systems," *Proc. Priv. Enhancing Technol.*, 2024.

[13] C. Mazzocca, A. Acar, S. Uluagac, R. Montanari, P. Bellavista, and M. Conti, "A survey on decentralized identifiers and verifiable credentials," *arXiv:2402.02455*, 2024.

[14] T. Feng and W. He, "Research on privacy preserving of searchable encryption," in *Proc. 2nd High Performance Computing and Cluster Technologies Conference*, 2018.

[15] G. S. Poh, J.-J. Chin, W.-C. Yau, K.-K. R. Choo, and M. S. Mohamad, "Searchable symmetric encryption: Designs and challenges," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, pp. 1–37, 2017.

[16] P. N. Mahalle, G. Shinde, and P. M. Shafi, "Rethinking decentralized identifiers and verifiable credentials for the internet of things," in *Internet of Things, Smart Computing and Technology: A Roadmap Ahead*, 2020.

[17] S. Wachter, "Normative challenges of identification in the internet of things: Privacy, profiling, discrimination, and the GDPR," *Comput. Law Secur. Rev.*, vol. 34, no. 3, 2018.

[18] R. N. Zaeem and K. S. Barber, "The effect of the GDPR on privacy policies: Recent progress and future promise," *ACM Trans. Manag. Inf. Syst.*, vol. 12, no. 1, 2020.

[19] F. Cirillo, G. Solmaz, E. L. Berz, M. Bauer, B. Cheng, and E. Kovacs, "A standard-based open source IoT platform: FIWARE," *IEEE Internet Things Magazine*, vol. 2, no. 3, 2019.

[20] P. Chaudhari and M. L. Das, "Privacy preserving searchable encryption with fine-grained access control," *IEEE Trans. Cloud Comput.*, vol. 9, no. 2, 2019.

[21] M. Li, S. Yu, N. Cao, and W. Lou, "Authorized private keyword search over encrypted data in cloud computing," in *Proc. 13th ACM Conference on Computer and Communications Security*, 2011.

[22] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," in *Proc. 13th ACM Conference on Computer and Communications Security*, 2006.

[23] H. Ren, H. Li, Y. Dai, K. Yang, and X. Lin, "Querying in internet of things with privacy preserving: Challenges, solutions and opportunities," *IEEE Network*, vol. 32, no. 6, 2018.

[24] S. De Capitani di Vimercati, S. Foresti, and P. Samarati, "Protecting data and queries in cloud-based scenarios," *SN Comput. Sci.*, vol. 4, no. 5, 2023.

[25] R. A. Dolin, "Search query privacy: The problem of anonymization," in *2010 AAAI Spring Symp. Series*, Mar. 2010.

[26] D. Cash, P. Grubbs, J. Perry, and T. Ristenpart, "Leakage-abuse attacks against searchable encryption," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, Oct. 2015, pp. 668–679.

[27] FIWARE, "FIWARE Official Documentation," February 2025. [Online]. Available: https://www.fiware.org/

[28] M. Aljumaili, R. Karim, and P. Tretten, "Metadata-based data quality assessment," *VINE J. Inf. Knowl. Manag. Syst.*, vol. 46, no. 2, 2016.