Working Paper presented at the

Peer-to-Peer Financial Systems
2025 Workshop

2025

Benchmarking Decentralized Identity Systems: A Multi-Dimensional Evaluation

Abylay Satybaldy

Exponential Science

Benchmarking Decentralized Identity Systems: A Multi-Dimensional Evaluation

***Abstract*—Decentralized Identity (DID) systems enable individuals to manage digital identifiers and credentials without relying on centralized authorities. As adoption accelerates, DID methods implemented on Distributed Ledger Technologies (DLTs) have gained prominence, each adopting distinct architectural and design strategies. However, rigorous cross-platform evaluations remain scarce. This study presents a comparative empirical analysis of three major DLT-based DID implementations: Ethereum (Ethr-DID), Hedera (HCS-DID), and XRP Ledger (XLS-40d). Using reference Software Development Kits (SDKs) and high-resolution benchmarking tools, we measure the latency, transaction cost, and privacy leakage of core DID operations under standardized experimental conditions. In addition to reporting absolute performance metrics, we normalize latency and cost relative to each platform's block time and native transfer fee. Privacy leakage is quantified using the Metadata-Leakage Score (MLS), an entropy-based metric that captures the amount of identifiable information exposed per operation. Results show that Ethereum incurs the highest latency and fees but aligns closely with its block time, indicating efficient integration with its smart contract layer. XRP Ledger (XRPL) offers consistent, low-cost performance, but suffers from higher metadata leakage due to verbose payload structures. Hedera achieves the fastest operation times and low fees, along with the lowest MLS, although its SDK introduces occasional processing variance. These findings demonstrate that DLT architecture and SDK design play a critical role in shaping the practical performance of DID systems. The study offers actionable insights for selecting appropriate DID stacks based on latency sensitivity, cost constraints, and privacy requirements.**

***Index Terms*—decentralized identity, distributed ledger technology, performance analysis**

I. Introduction

Decentralized Identity (DID) has emerged as a transformative approach to managing digital identities without reliance on centralized authorities. By prioritizing user autonomy, the DID model enables individuals to own, manage, and selectively share their credentials [1]. Unlike traditional identity systems that depend on centralized intermediaries for credential issuance and verification, DID systems eliminate single points of failure and enhance user privacy through cryptographically verifiable identifiers.

Conventional identity management relies on siloed databases maintained by governments, corporations (e.g., Google, Facebook), or federated identity providers such as OpenID Connect. While these systems support interoperability via single sign-on, they introduce privacy and surveillance risks by allowing identity providers to track user activity across services [2], [3].

As a partial alternative, crypto wallet addresses offer self-generated public-private key pairs for authentication without third-party involvement. While this improves autonomy, wallet-based identity lacks support for expressing verifiable claims or associating identifiers with rich, real-world attributes [4].

DID frameworks address these limitations by combining cryptographic trust mechanisms with interoperable standards for expressing and verifying identity data. Core components include World Wide Web Consortium (W3C) Decentralized Identifiers (DIDs) [5] and Verifiable Credentials (VCs) [6], which allow entities to create resolvable, tamper-evident identifiers and attach digitally signed credentials. DIDs resolve to DID Documents hosted on decentralized networks (e.g. blockchains or InterPlanetary File System (IPFS)), containing public keys, service endpoints, and metadata for authentication and key rotation.

Importantly, DIDs support formal trust frameworks through native compatibility with cryptographic proof formats such as JSON Web Tokens (JWTs) and Linked Data Proofs [7], [8]. This enables secure issuance and verification of credentials across diverse ecosystems, going beyond the authentication capabilities of crypto wallets.

As adoption grows, various DLT platforms have introduced their own DID methods, each with distinct architectural decisions. Throughout this paper, we use the term "blockchain" interchangeably with DLT, noting that blockchain is a specific implementation within the broader DLT landscape. For real-world viability, DID systems must balance responsiveness, cost, and privacy—yet systematic, empirical comparisons of these trade-offs remain scarce.

To address this gap, we conduct a comparative evaluation of three widely recognized DLT-based DID reference implementations: Ethereum (Ethr-DID), Hedera (HCS-DID), and the XRP Ledger (XLS-40d). These platforms were selected based on technical maturity, adoption in industry, and alignment with W3C standards. We focus on three critical dimensions of system performance: latency, transaction cost, and on-chain metadata leakage—each vital for scalable, privacy-aware identity deployments.

This study is guided by the following research questions:

- **RQ1 – Performance Characterization:** What are the absolute and relative latency, transaction cost, and metadata-leakage profiles of core DID operations on the selected DLT platforms under standardized benchmark conditions?
- **RQ2 – Architectural Attribution:** How do the architectural features and SDK workflows of each platform explain the performance outcomes observed in RQ1?
- **RQ3 – Deployment Trade-offs:** Which platform offers the most effective balance among responsiveness, cost-efficiency, and privacy for real-world DID applications?

By addressing these questions, our study advances the field from qualitative comparisons to a structured, data-driven evaluation that informs both researchers and practitioners in selecting suitable DID platforms based on deployment-specific priorities.

The remainder of this paper is organized as follows: Section II reviews prior research on decentralized identity systems. Section III introduces key architectural components of DIDs and their reference implementations. Section IV describes our evaluation methodology, including metrics and experimental setup. Section V presents results and analysis, followed by conclusions and future research directions in Section VII.

II. Related Work

Several survey papers have examined the landscape of decentralized identity and Self-Sovereign Identity (SSI) systems. These works typically introduce foundational technologies such as DIDs and VCs, and discuss a wide range of identity systems and their underlying design choices. Notably, Dunphy and Petitcolas [9] provided one of the earliest analyses of blockchain-based identity management, while Mühle et al. [10] outlined the core components of SSI architectures. More recently, Krul et al. [11] systematized knowledge in the field by evaluating trust assumptions and requirements across various SSI implementations.

Comparative analyses of existing DID platforms have also emerged. These studies frequently focus on solutions such as uPort and Hyperledger Indy, evaluating their design and dependencies based on publicly available documentation. For instance, Satybaldy et al. [12] introduced a structured evaluation framework and applied it to several SSI platforms. In parallel, researchers have examined the architectural building blocks of blockchain-based identity systems. Dib et al. [13] conducted a detailed investigation of decentralized identity architectures, highlighting technical challenges and recommending architectural improvements.

While these studies offer valuable insights into the conceptual and architectural foundations of decentralized identity, they often lack empirical evaluation. Specifically, few works provide a quantitative comparison of DID solutions focused on the performance of core operations, latency characteristics, or transaction cost under controlled experimental conditions. Most prior analyses remain qualitative or conceptual, limiting their utility for benchmarking or deployment planning.

In a separate line of research, performance benchmarking of DLT platforms has been explored. Amherd et al. [14] investigated the transaction throughput and efficiency of Hedera Hashgraph, while other studies have measured the performance of Ethereum and XRPL [15]–[17]. However, these analyses do not specifically assess the performance of DID systems or compare identity functionality across DLTs.

This paper addresses this gap by presenting a focused, comparative, and empirical evaluation of DID reference implementations on Ethereum, Hedera, and XRPL. By benchmarking key DID operations, examining platform-specific architectural characteristics, and quantifying performance in terms of latency, cost, and metadata leakage, this work provides practical insights for researchers and developers evaluating DLTs for decentralized identity deployment. Our contribution extends prior conceptual analyses with a data-driven, performance-oriented methodology grounded in real SDK implementations and repeatable benchmarking.

III. Background

A. Decentralized Identifiers (DIDs)

DIDs are standardized by the W3C and offer a novel, decentralized approach to digital identification within the broader DID framework [5]. DIDs follow the URI format:

```
did:<method>:<method-specific-identifier>
```

Here, `did` denotes the scheme, `method` indicates the specific DID method (e.g., did:web, did:ion, did:ethr), and `method-specific-identifier` is a unique identifier typically derived from cryptographic material.

B. DID Document

DIDs are resolvable to DID Documents, which are JSON-based data structures containing public keys, verification methods, and service endpoints. These documents enable authentication, credential issuance, and other trust-related operations. They can be serialized in JSON or JSON for Linked Data (LD) format, ensuring broad interoperability [18]. Depending on the DID method, the DID Document may be stored:

- **On-chain:** Stored directly on a blockchain or distributed ledger.
- **Off-chain:** Stored externally (e.g., in IPFS or a database) with on-chain references.

An example DID Document:

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:example:123456789abcdefghi",
  "verificationMethod": [
    {
      "id": "did:example:123456789abcdefghi#key-1",
      "type": "Ed25519VerificationKey2018",
      "controller": "did:example:123456789abcdefghi",
      "publicKeyBase58": "GfH345jHK...asd67X"
    }
  ],
  "authentication": [
    "did:example:123456789abcdefghi#key-1"
  ],
  "assertionMethod": [
    "did:example:123456789abcdefghi#key-1"
  ],
  "service": [
    {
      "id": "did:example:123456789abcdefghi#vcs",
      "type": "VerifiableCredentialService",
      "serviceEndpoint": "https://example.com/vcs"
    }
  ]
}
```

A DID Document contains various properties, including public keys, authentication methods, and service endpoints, as shown in Table I.

Property	Description
`@context`	Defines the JSON-LD schema for DIDs.
`id`	Globally unique DID for the entity.
`verificationMethod`	Public keys for verifying cryptographic signatures.
`authentication`	Keys used for authentication.
`assertionMethod`	Keys used to issue Verifiable Credentials.
`service`	Service endpoints for identity-linked services.

TABLE I: DID Document components

C. DID Method

A DID Method defines how DIDs are created, resolved, updated, and deactivated within a specific system or network. Each method must specify procedures for core DID operations shown in Table II. DID resolution is performed by DID Resolvers, which interpret the method and fetch the document from the respective storage backend.

D. Decentralized Identity on Hedera

Hedera provides a robust foundation for decentralized identity by combining its high-throughput, low-latency consensus mechanism with a DID framework aligned with emerging W3C standards. This section outlines the Hedera DID method, supported operations, and available tooling.

1) Hedera DID Method Design: Hedera's approach leverages the Hedera Consensus Service (HCS) to manage DIDs and DID Documents in a tamper-evident and transparent way. The DID method records DID-related events—such as creation, updates, and deletions—on HCS. These events include metadata and references, rather than the full DID document content. Off-chain storage is used for the actual documents, while HCS holds pointers (e.g., hashes) to retrieve them. This design offers an efficient and scalable architecture.

According to the Hedera DID Method Specification [19], the format is:

`did:hedera:<network>:<identifier>_<topicId>`

Here, `network` is the Hedera network (e.g., mainnet, testnet), `identifier` is a Base58-encoded DID root public key, and `topicId` refers to the HCS Topic ID where events are recorded. For example, Topic ID `0.0.29656231` refers to topic number `29656231` in shard `0`, realm `0`.

Each DID Document must include a public key of type `Ed25519VerificationKey2018` and support standard verification mechanisms such as authentication, assertion, key agreement, and capability delegation.

2) DID Operations: Hedera supports the full set of DID operations via HCS messages, as shown in Table II. Resolution occurs through a Hedera Mirror Node (HMN). Create, Update, Revoke, and Delete operations are submitted via HCS messages (Figure 1).

3) Reference Implementation: The Hedera DID SDK for JavaScript (`did-sdk-js`) [20] provides a reference implementation. It allows developers to create and manage DIDs and VCs via HCS.

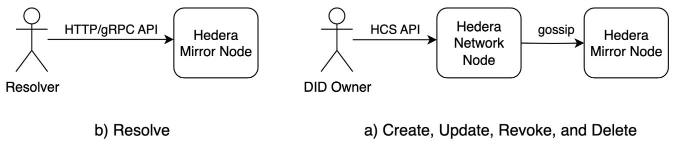


Fig. 1: Hedera DID Operations [19]

E. Decentralized Identity on Ethereum

Ethereum supports decentralized identity using smart contracts, particularly through the ERC-1056 standard. This framework enables self-sovereign identity without centralized intermediaries, suitable for applications in DeFi, Web3 authentication, and more.

1) Ethereum DID Method Design: Ethereum's DID method is implemented via the Ethr-DID Method [21], which uses the ERC-1056 standard. The format is:

`did:ethr:<network>:<identifier>`

Here, `network` denotes the Ethereum chain (e.g., mainnet, sepolia), and `identifier` is an Ethereum address (0x...) or HEX-encoded secp256k1 public key.

Each identifier is self-controlled by its associated address. Controllers can delegate or transfer control, including to smart contracts (e.g., multi-signature wallets). DID Documents are stored off-chain but secured via the ERC-1056 smart contract, which records integrity and controller information. Verification is done using cryptographic methods (`secp256k1`) or smart contract logic.

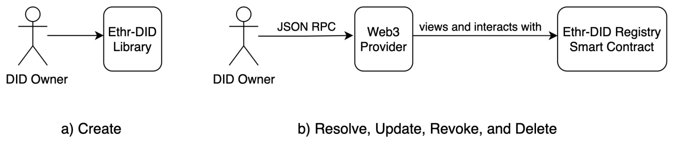


Fig. 2: Ethereum DID operations.

2) DID Operations: Ethereum supports the full set of DID operations through ERC-1056 transactions, as shown in Table II. The DID is created off-chain by generating a new Ethereum address (Figure 2a). Resolve, Update, Revoke, and Delete operations require interaction with the smart contract (Figure 2b).

3) Reference Implementation: The Ethr-DID Library [22] is the official reference maintained by the Decentralized Identity Foundation (DIF). It integrates with frameworks such as Veramo [23] and supports full DID lifecycle management.

F. Decentralized Identity on XRP Ledger

The XRPL introduced native DID support via XRP Ledger Standard (XLS-40), which was activated on October 30, 2024 [24]. This implementation aligns with W3C DID standards and enables on-ledger identity.

Operation	Description	Hedera	Ethereum	XRPL
DID Creation	Establishes a new DID and its corresponding DID Document.	A new DID is registered by submitting a `ConsensusSubmitMessage` transaction to an HCS topic.	DIDs are implicitly created when a new Ethereum address is generated; the ERC-1056 contract enables DID document management.	A `DIDSet` transaction creates a DID entry with its associated DID Document or metadata.
DID Resolution	Retrieves the DID Document associated with a DID.	Query the relevant HCS topic via a mirror node to reconstruct the DID Document.	Dynamically reconstructed by querying the smart contract logs and state variables.	Use `LedgerEntry` or account objects to fetch the stored DID object.
DID Update	Alters elements of the DID Document, including keys, authentication methods, or service endpoints.	Event payloads specify additions or modifications to objects (e.g., `Service`, `VerificationMethod`).	Controllers update attributes or manage delegates using smart contract functions.	Submit another `DIDSet` transaction to update existing attributes.
DID Revocation	Removes selected properties from the DID Document, such as keys or services.	Objects to be removed are indicated in the event payload.	Keys or attributes are revoked through `DIDDelegateChanged` or `DIDAttributeChanged` events.	Not explicitly supported; attributes may be cleared or replaced via `DIDSet`.
DID Deactivation	Permanently disables the DID, making it non-resolvable.	Entire DID Document is nullified, rendering the DID inactive.	Setting the controller to the null address (`0x0`) deactivates the DID.	A `DIDDelete` transaction deactivates the DID and reclaims reserved XRP.

TABLE II: Comparison of core DID operations across Hedera, Ethereum, and XRPL.

1) XRPL DID Method Design: XRPL's approach uses ledger-native DID objects and transactions, avoiding the need for smart contracts. It defines two transaction types: `DIDSet` and `DIDDelete` [25].

The decentralized identifier format is:

`did:xrpl:<network-id>:<identifier>`

Where `network-id` identifies the XRPL network (e.g., 1 for Mainnet), and `identifier` is the Account ID or hex-encoded master public key.

2) DID Operations: The set of DID operations supported by XRPL is summarized in Table II.

3) Reference Implementation: The XRPL SDK for JavaScript [26] offers a native reference for interacting with DID features on the XRP Ledger.

IV. Methodology

This research proposes a systematic benchmarking framework to evaluate DID reference implementations across various DLTs. The framework targets both fundamental DID operations and complete identity management workflows, as summarized in Table II. These operations represent the lifecycle events essential to decentralized identity systems, such as creation, resolution, update, revocation, and deactivation of DIDs.

A. Reference Implementations

The study evaluates representative DID implementations across multiple DLT platforms. To ensure interoperability and comparability, we focus on DID methods that comply with the W3C DID specification. The evaluation adopts a neutral, technology-agnostic stance.

Reference implementations were selected based on public indicators of adoption and maturity, including GitHub star counts, repository activity, and developer community engagement. To ensure consistency and eliminate variability due to differences in SDK languages, all tests were performed using JavaScript-based SDKs. This design choice facilitates uniformity in benchmarking environments, tooling, and execution.

Table III summarizes the DID SDKs evaluated for each DLT platform.

DLT	DID Method	SDK	Version	Language	Chain
Ethereum	did:ethr [21]	Ethr-DID SDK [27]	2.1.2	JavaScript	Sepolia
Hedera	did:hedera [19]	Hedera DID SDK [20]	0.1.1	JavaScript	Hedera Testnet
XRPL	did:xrpl [24]	XRPL SDK [26]	2.4.1	JavaScript	XRPL Testnet

TABLE III: DID SDKs across DLTs

B. Evaluation Metrics

The benchmarking framework evaluates DID implementations across three key dimensions: performance, cost efficiency, and privacy. Table IV summarizes the core metrics used in the analysis.

Category	Metric	Description
Performance	Latency	Measures the time elapsed from initiation to completion of each DID operation.
Cost Efficiency	On-chain operation cost	Quantifies transaction fees in both native tokens and USD to evaluate economic efficiency.
Privacy	On-chain metadata leakage	Quantifies the amount of identifying or fingerprintable information disclosed via on-chain metadata per operation.

TABLE IV: Key Metrics for DID SDK Benchmarking

1) Full Cycle Operation: To capture the end-to-end performance of a realistic DID lifecycle, we introduce a composite metric termed the `Full Cycle Operation`. This metric aggregates the measured latency and cost of the core lifecycle actions—Create, Resolve, Update, Revoke, and Delete. The Full Cycle model thus provides a holistic view of cumulative performance and cost, complementing the insights from isolated benchmarks.

2) Relative Latency and Cost Metrics: To facilitate fair comparison across DLTs with heterogeneous network properties, we normalize latency and cost using platform-specific baselines. For latency, the `relative latency` L_{rel} is defined as:

$$L_{\text{rel}} = \left(\frac{\bar{L}_{\text{op}}}{\bar{T}_{\text{block}}} \right) \times 100\%$$

where $\bar{L}_{\text{op}}$ is the mean latency of a DID operation, and $\bar{T}_{\text{block}}$ is the average block time (or consensus event interval) of the platform [28]. This metric contextualizes responsiveness relative to native block production rates.

For cost, the `relative cost` C_{rel} compares the mean fee of a DID operation $\bar{C}_{\text{op}}$ to the cost of a standard token transfer $\bar{C}_{\text{std}}$:

$$C_{\text{rel}} = \left(\frac{\bar{C}_{\text{op}}}{\bar{C}_{\text{std}}} \right) \times 100\%$$

These relative metrics reveal the operational overhead of DID interactions and allow cross-platform evaluation independent of absolute performance or fee levels.

C. Benchmarking Tools and Hardware Setup

For each JavaScript-based DID SDK, benchmarking was conducted using consistent measurement tools and controlled environments:

- **Latency:** Measured using Node.js high-resolution performance timers.
- **Cost:** On-chain transaction costs were retrieved via ethers.js and blockchain explorer APIs, then converted to USD using contemporaneous market prices.

Experiments were conducted on an Amazon EC2 c5.4xlarge instance (16 vCPUs, Intel Xeon Platinum @ 3.0 GHz, 32 GiB RAM), running Amazon Linux 2023. This setup ensures computational consistency and supports the reproducibility of results in future studies.

D. Experimental Design

The benchmarking experiment was designed as a single, unified evaluation of DID SDKs across three DLT platforms. All SDKs were tested under identical conditions—using the same benchmarking tools, hardware environment, and sequence of operations. For each SDK, a single experiment consisting of 100 iterations per DID operation was conducted to ensure statistical reliability. During each iteration, latency, transaction cost, and on-chain metadata were recorded to compute the corresponding performance, cost-efficiency, and privacy metrics. This controlled experimental design ensures consistency across platforms and isolates the effects of the underlying technology stacks. The results reported throughout the paper are derived from this singular experimental setup.

V. Results and Discussion

This section presents a comparative analysis of core DID operations across three DLT platforms: Ethereum (Ethr-DID), XRP Ledger (XLS-40d), and Hedera (HCS-DID). The evaluation integrates empirical performance measurements with structural insights into the operational workflows implemented by each reference SDK. This dual approach enables both quantitative benchmarking and qualitative comparison of architectural design choices, offering a comprehensive perspective on how each platform supports decentralized identity functionality.

A. Latency Analysis

Figure 3 presents the latency distributions for five core DID operations along with the composite Full Cycle operation across Ethereum, XRPL, and Hedera. Latencies are measured in seconds and presented in separate subplots for each platform. For context, a dashed green line in each plot indicates the mean block interval time for Ethereum (12.06s), XRPL (3.87s), and Hedera (2.90s) [29], allowing for direct comparison between operation latency and standard transaction processing intervals. Figure 4 complements this by presenting the relative latency of each operation as a percentage of the respective platform's block time. For Hedera, which operates on a Hashgraph rather than a blockchain, the mean consensus time is used instead. Detailed statistics are available in Appendix Table IX.

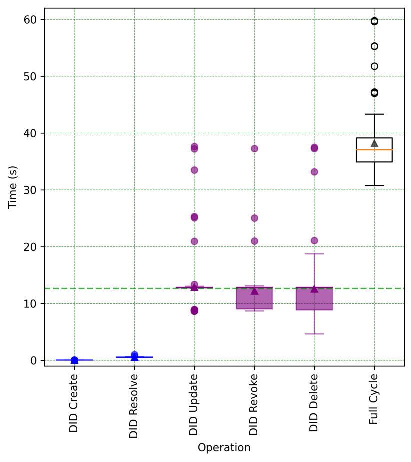
Across all platforms, DID Resolve consistently demonstrates the lowest latency (Figure 3). This outcome is expected, as resolution is an off-chain operation that retrieves existing data rather than submitting new transactions. Hedera records the fastest resolve time with a mean latency of 0.056s. XRPL follows at 0.076s, and Ethereum achieves 0.534s. The relatively fast and consistent results are due to the read-only nature of resolve operations and the efficiency of data retrieval mechanisms across the three platforms.

In contrast, on-chain operations show greater latency variation. Ethereum's DID Create operation is implemented off-chain, resulting in an exceptionally low latency of 0.011s. However, its Update, Revoke, and Delete operations require on-chain execution, with mean latencies between 12.2 and 12.6 seconds. These align closely with Ethereum's block production interval, as shown in Figure 4. Notably, these operations display significant variability, with standard deviations exceeding 3–4 seconds and maximum latencies reaching up to 37 seconds.

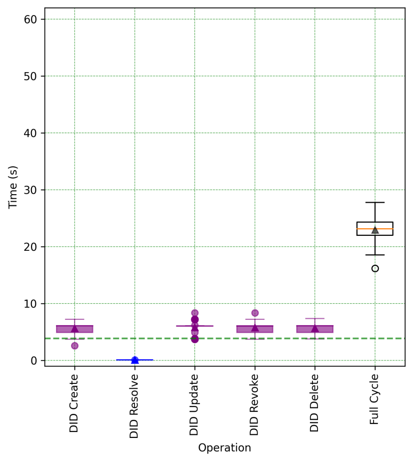
XRPL exhibits higher relative latencies for on-chain operations. As illustrated in Figure 3b, Create, Update, Revoke, and Delete operations average between 5.6 and 5.8 seconds. These values correspond to 145–150% of XRPL's block interval, indicating that DID operations incur additional overhead compared to standard transactions. However, XRPL's performance remains consistent, with low variance across samples.

Hedera offers the fastest on-chain operation times overall, as shown in Figure 3c. Create, Update, Revoke, and Delete operations show mean latencies between 4.0 and 4.4 seconds—equivalent to 137–151% of Hedera's mean consensus time. Although moderate overhead exists due to SDK-side processes such as mirror node subscription and message parsing, Hedera still outperforms Ethereum and XRPL in absolute terms. The highest variability appears in the Create operation, which depends on asynchronous confirmation of HCS-published events.

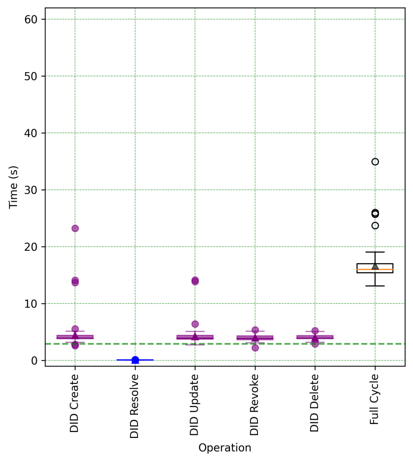
In summary, while Ethereum shows the highest absolute latencies, its DID operations scale proportionally with block time, reflecting efficient integration with its smart contract layer and minimal SDK-induced delays. Hedera delivers the best overall performance in terms of speed, while XRPL excels in latency consistency. However, both Hedera and XRPL reveal



(a) Latency of Operations for Ethereum.



(b) Latency of Operations for XRPL.



(c) Latency of Operations for Hedera.

Fig. 3: Latency distribution of DID operations across Ethereum, XRPL, and Hedera. Purple data points represent on-chain operations, while blue points denote off-chain operations. Each subplot includes the respective network's mean block time, indicated by a dashed green line.

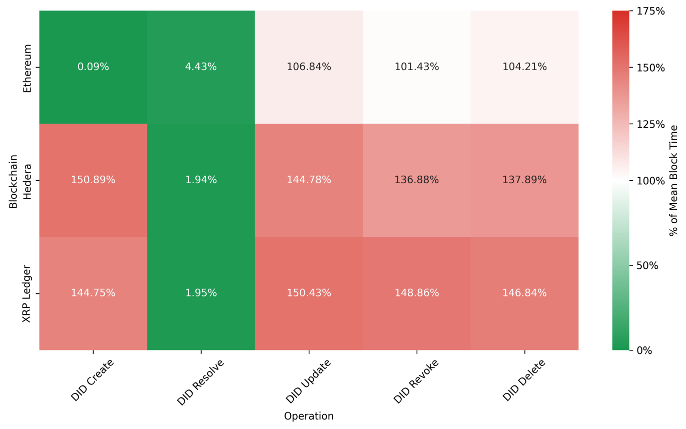


Fig. 4: Relative latency of DID operations, expressed as a percentage of the platform's mean block time. This normalised metric allows comparison of operational delays across DLTs with differing consensus speeds (Ethereum: 12.06s, XRPL: 3.87s, Hedera: 2.90s).

elevated relative latencies compared to their block or consensus times due to additional SDK-level processing and validation steps, which will be further examined in Section V-C.

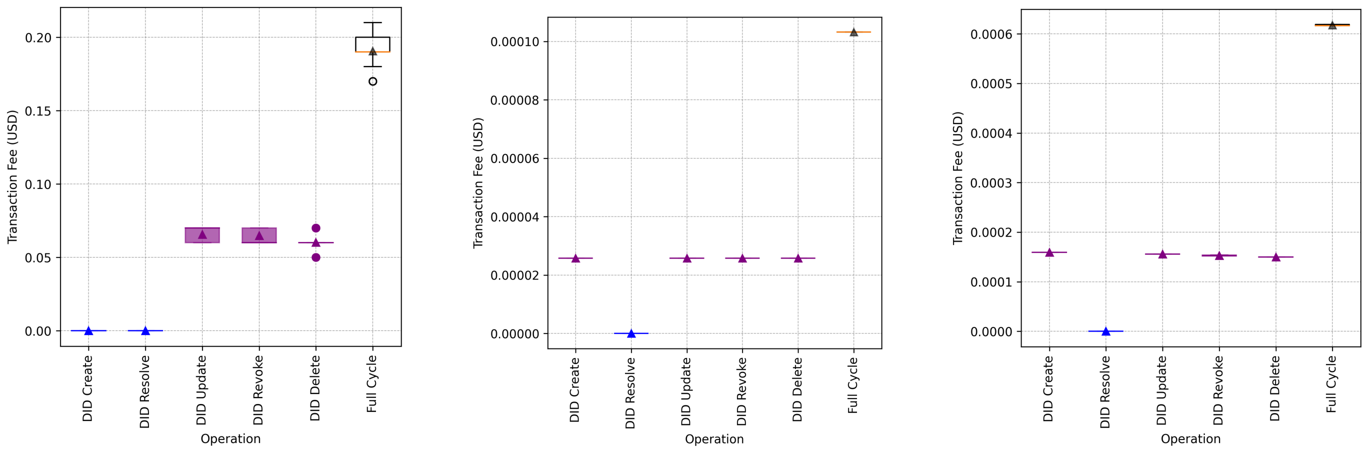
B. Transaction Cost Analysis

Figure 5 presents the cost distribution of DID operations, and Full Cycle across Ethereum, XRPL, and Hedera. Transaction costs are reported in USD, estimated based on market prices as of April 15, 2025. In alignment with the latency analysis, each subplot includes a dashed green line representing the mean cost of a standard cryptocurrency transfer on the corresponding blockchain (Ethereum: \$0.04 at 1.2 Gwei, XRPL: \$0.000021, Hedera: \$0.0001). This enables direct comparison between DID operation fees and typical network transaction fees. To ensure consistency in evaluation, the same gas price (1.2 Gwei) was applied to both Ether transfer transactions and our Ethereum-based DID experiments. Detailed cost statistics can be found in the Appendix - Table X.

As shown in Figure 5a, Ethereum incurs the highest transaction costs for on-chain DID operations. DID Update, Revoke, and Delete average \$0.066, \$0.065, and \$0.060, respectively. Meanwhile, DID Create and Resolve operations incur no cost because they are off-chain in the Ethr-DID model. The Full Cycle operation reaches an average of \$0.19. These values are approximately 150–164% higher than Ethereum's standard Ether transfer cost of \$0.04 (Figure 6). This disparity stems from the nature of DID operations, which require interaction with the ERC-1056 smart contract. Specifically, each update, revoke, or delete operation triggers a state change and emits one or more events to the Ethereum log system. This results in higher gas usage due to storage writes, event indexing, and signature verification overhead, which are substantially more resource-intensive than basic transfers.

The XRP Ledger shows extremely low and consistent transaction costs across all DID operations (Figure 5b). The mean cost for Create, Update, Revoke, and Delete operations is consistently \$0.000021—matching the standard XRP payment fee. The DID Resolve operation incurs no cost. The current minimum transaction fee required by the XRPL network is 0.00001 XRP (10 drops) [30], and all DID operations align with this threshold. The cost distribution is exceptionally narrow with zero variance, confirming that XRPL maintains perfect fee predictability and minimal cost. This reflects XRPL's lightweight transaction model and the fact that DID operations are implemented as standard transaction types.

Hedera also demonstrates low-cost DID operations (Figure 5c). The mean cost for DID Create is \$0.00016, while Update, Revoke, and Delete average around \$0.00015. The Full



(a) Cost of DID Operations for Ethereum.

(b) Cost of DID Operations for XRPL.

(c) Cost of DID Operations for Hedera.

Fig. 5: Cost distribution of DID operations across Ethereum, XRPL, and Hedera (in USD, based on prices as of April 15, 2025). Purple data points represent on-chain operations, while blue points denote off-chain operations. Each subplot includes a dashed green line indicating the mean cost of a standard cryptocurrency transfer on the respective network.

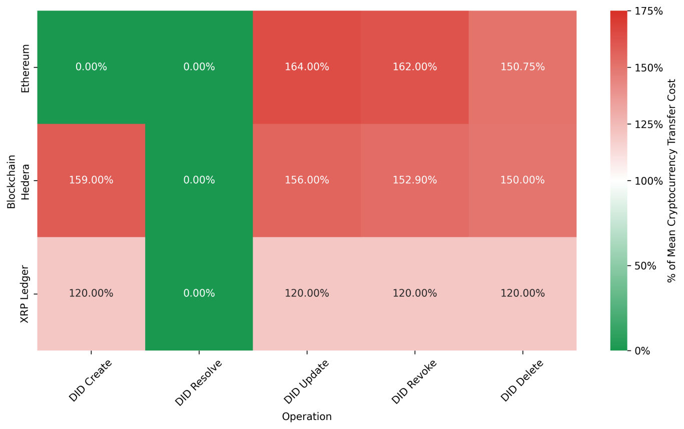


Fig. 6: Relative transaction cost of DID operations, expressed as a percentage of the mean fee for a standard cryptocurrency transfer on each platform. This baseline-adjusted metric enables cross-platform cost comparison.

Cycle totals approximately \$0.00062. Compared to Hedera's standard `CryptoTransfer` fee of \$0.0001 [31], DID operations are roughly 150–160% more expensive (Figure 6). Although DID operations on Hedera use the lightweight `ConsensusSubmitMessage` service, their transaction cost is higher than that of basic transfers. This is primarily due to the structured and signed JSON payloads required by the DID method, as well as additional metadata and formatting overhead introduced by the DID SDK.

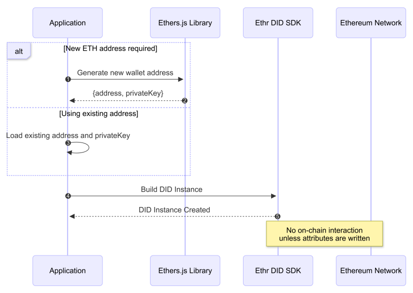
In summary, Ethereum remains the most expensive platform for DID operations, largely due to the use of gas-intensive smart contract calls and log emissions. XRPL achieves the lowest and most consistent costs, maintaining parity with its baseline fee structure. Hedera offers low fees with slight overhead driven by DID-specific data formatting and SDK integration. These results highlight the importance of cost efficiency in the scalability of decentralized identity systems, with XRPL and Hedera emerging as more suitable options for high-volume identity use cases.

C. *Design-Level Analysis and Performance Interpretation*

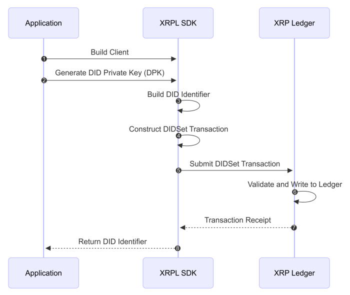
This section addresses Research Question 2 (RQ2). Building on the empirical findings from latency and cost analysis, we examine the design-level factors—such as transaction models, consensus mechanics, and SDK architecture—that shape the operational characteristics of each DID implementation. By connecting technical design patterns to performance outcomes, we provide a deeper interpretation of why certain platforms show faster execution, greater consistency, or lower costs.

1) DID Create Operation: The Ethr-DID implementation follows an implicit registration model, where a DID is derived from an Ethereum address without requiring interaction with the blockchain. As illustrated in the sequence diagram (Figure 7a), the creation process simply involves generating a key pair and instantiating a DID object in memory using the Ethr-DID SDK. When a new key is needed, it is generated locally, for example, via the `ethers.js` library. Since no transaction is submitted to the network, the process avoids consensus delays entirely. Consequently, Ethereum achieved the lowest mean latency of approximately 0.011 seconds, with minimal variance, highlighting the performance benefits of off-chain DID instantiation.

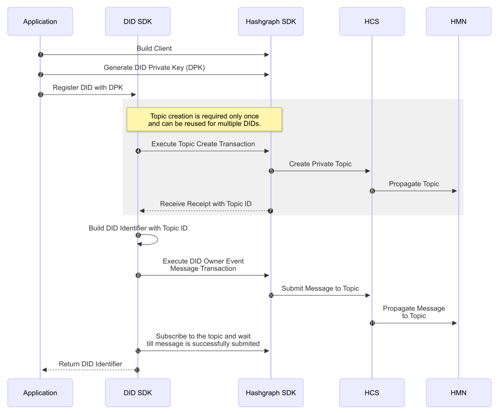
In contrast, XRPL mandates an on-chain registration through a `DIDSet` transaction (Figure 7b). This includes generating a key pair, constructing and submitting the transaction, and waiting for ledger consensus and validation. The process introduces unavoidable delays due to block finality and SDK processing. The mean latency was measured at approximately 5.6 seconds, with moderate variance. While performance remains stable, the need for network confirmation results in



(a) Ethereum DID Create

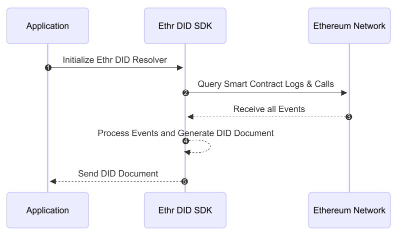


(b) XRPL DID Create

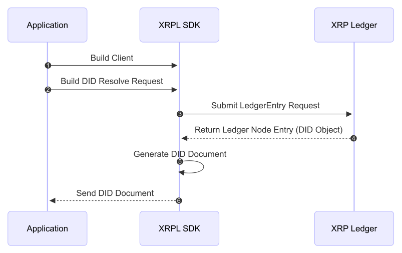

Application
DID SDK
Hashgraph SDK
HCS
HMN
Build Client
Generate DID Private Key (DPK)
Register DID with DPK
Topic creation is required only once and can be reused for multiple DIDs.
Execute Topic Create Transaction
Create Private Topic
Propagate Topic
Receive Receipt with Topic ID
Build DID Identifier with Topic ID
Execute DID Owner Event Message Transaction
Submit Message to Topic
Propagate Message to Topic
Subscribe to the topic and wait till message is successfully submited
Return DID Identifier
Application
DID SDK
Hashgraph SDK
HCS
HMN

(c) Hedera DID Create

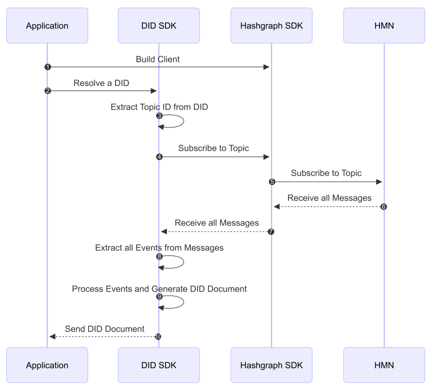
Fig. 7: DID Create operation sequence diagram.



(a) Ethereum DID Resolve



(b) XRPL DID Resolve



(c) Hedera DID Resolve

Fig. 8: DID Resolve operation sequence diagram.

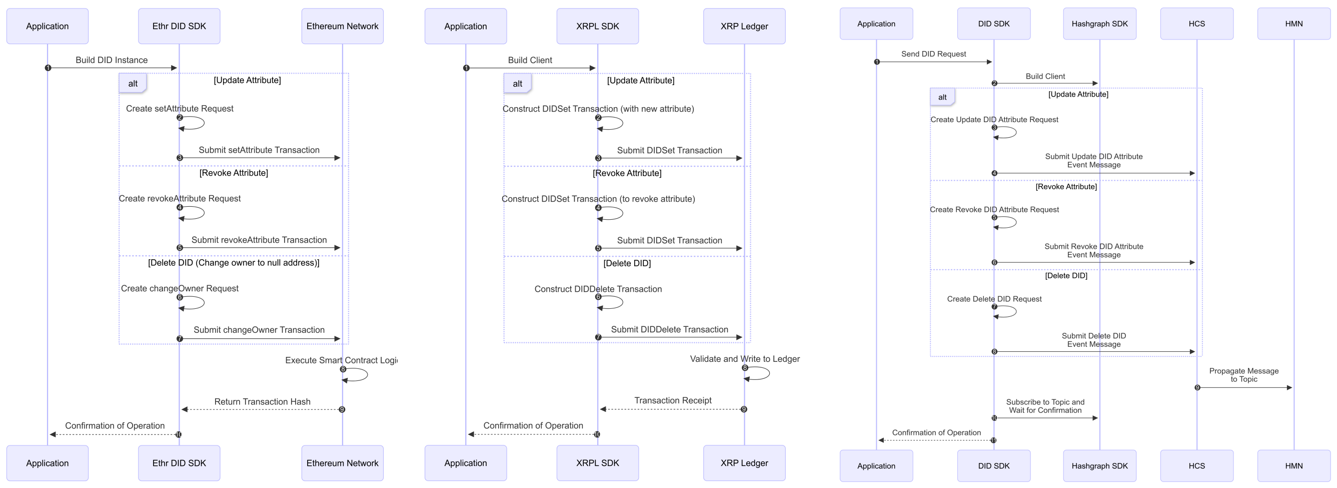
significantly higher latency compared to Ethereum's off-chain model.

Hedera's HCS-DID method involves a more complex sequence compared to other platforms (Table V). DID creation requires the initialization of a new HCS topic, the publication of a DID owner event message, and confirmation via subscription to the topic through mirror nodes. The mean latency for this process is approximately 7 seconds. However, since the Hedera DID method supports the creation of multiple DIDs under a single topic ID, we adjusted our benchmarking setup to initialize a single HCS topic and reuse it for multiple DID registrations—aligning the setup more closely with Ethereum and XRPL. When excluding the topic creation step, the average latency for DID creation on Hedera decreases to approximately 4.3 seconds, though it shows greater variance and more frequent outliers, reaching up to 23 seconds. This variability is primarily due to the multi-step process, including event propagation across the HMN and confirmation delays handled by the SDK. Additional overhead is introduced by the combined use of the Hashgraph SDK and the DID SDK, as shown in Figure 7c.

2) DID Resolve Operation: The observed differences in DID Resolve latency across Ethereum (0.534s), XRPL (0.076s), and Hedera (0.056s) can be attributed to the underlying design and data retrieval mechanisms illustrated in their respective sequence diagrams in Figure 8. Ethereum's Ethr-DID resolver must query the ERC-1056 smart contract for logs and state variables, requiring multiple read operations across potentially deep historical logs to reconstruct the DID document. This dependency on smart contract event enumeration and state inspection contributes to longer latency. XRPL resolves DIDs by performing a single `LedgerEntry` request to fetch a pre-structured DID object stored on-chain, resulting in faster retrieval, though still constrained by ledger lookup and validation. In contrast, Hedera's design leverages a lightweight event stream model where all DID-related messages are sequentially

TABLE V: Latency and Design Trade-offs for DID Creation

Factor	Ethereum	XRPL	Hedera
On-chain Write Required	No	Yes	Yes
Consensus Interaction	None	Single transaction	Two transactions
SDK Overhead	Minimal	Moderate	High
Latency Outliers	Rare	Occasional	Frequent
Mean Latency	∼11 ms	∼5.6 s	∼4.3 s


Application
Ethr DID SDK
Ethereum Network
Build DID Instance
alt
[Update Attribute]
Create setAttribute Request
Submit setAttribute Transaction
[Revoke Attribute]
Create revokeAttribute Request
Submit revokeAttribute Transaction
[Delete DID (Change owner to null address)]
Create changeOwner Request
Submit changeOwner Transaction
Execute Smart Contract Logic
Return Transaction Hash
Confirmation of Operation
XRPL SDK
XRP Ledger
Build Client
Construct DIDSet Transaction (with new attribute)
Submit DIDSet Transaction
Construct DIDSet Transaction (to revoke attribute)
[Delete DID]
Construct DIDDelete Transaction
Submit DIDDelete Transaction
Validate and Write to Ledger
Transaction Receipt
DID SDK
Hashgraph SDK
HCS
HMN
Send DID Request
Create Update DID Attribute Request
Submit Update DID Attribute Event Message
Create Revoke DID Attribute Request
Submit Revoke DID Attribute Event Message
Create Delete DID Request
Submit Delete DID Event Message
Propagate Message to Topic
Subscribe to Topic and Wait for Confirmation

(a) Ethereum (b) XRPL (c) Hedera

Fig. 9: DID Update, Revoke, and Delete operation sequence diagrams.

published to a single HCS topic. The resolve process involves subscribing to that topic via the HMN and parsing the ordered messages to reconstruct the DID document. Because message retrieval is optimized through streaming APIs and topic state is isolated from global ledger history, Hedera achieves the lowest latency. Additionally, Hedera's event-driven model avoids costly log traversal or contract interactions, contributing to its high responsiveness in resolution operations.

3) DID Update, Revoke and Delete Operations: The latency performance of the DID Update, Revoke, and Delete operations varies significantly across Ethereum, XRPL, and Hedera, reflecting the underlying architectural and design choices of each network and their corresponding DID methods.

On Ethereum, these operations are executed through on-chain transactions that interact directly with the `EthrDIDRegistry` smart contract, as shown in Figure 9a. Specifically, attribute updates are handled via `setAttribute()`, revocations via `revokeAttribute()`, and deletion (ownership change) via `changeOwner()`. As each of these invokes a new transaction and requires confirmation through Ethereum's probabilistic finality mechanism, they are subject to the full latency of block confirmation and gas price dynamics. The mean latencies observed—12.9 seconds for Update, 12.2 seconds for Revoke, and 12.6 seconds for Delete—are the highest among the platforms studied. Furthermore, the wide standard deviations and long-tail maxima (up to 37 seconds) reflect Ethereum's volatile transaction inclusion times and congestion sensitivity. These results are consistent with the interaction-heavy nature of Ethereum's smart contracts and their reliance on EVM execution and block mining.

In XRPL, DID Update and Revoke operations are both executed through `DIDSet` transactions, while DID Delete uses a dedicated `DIDDelete` transaction. All three operations require submitting a transaction to the ledger and waiting for deterministic consensus validation, as illustrated in Figure 9b. Compared to Ethereum, XRPL demonstrates significantly lower latency, with mean times of 5.8 seconds (Update), 5.8 seconds (Revoke), and 5.7 seconds (Delete). This improvement can be attributed to XRPL's fast consensus protocol and deterministic finality, which avoids the uncertainty and delays associated with transaction confirmation in Ethereum. Nevertheless, because each operation still involves writing to the ledger and propagating changes across the network, a baseline latency is unavoidable. The relatively narrow variance across operations suggests a stable and predictable performance profile.

Hedera's HCS-DID method follows a distinct event-driven model. For all three operations the application submits a signed event message to a dedicated HCS topic, which is then propagated via the HCS and confirmed through the HMN infrastructure (Figure 9c). Despite Hedera's generally faster consensus finality (2.9s), the average latency for these operations (4.2 seconds for Update, 4.0 seconds for Revoke, and 4.0 seconds for Delete) is shaped by the asynchronous nature of topic message propagation and confirmation. Compared to Ethereum and XRPL, Hedera consistently demonstrates the lowest mean latency and variance. However, the presence of occasional outliers (with maxima reaching 14.1 seconds for Update) is explained by additional consensus-level interactions and SDK-induced delays—particularly in message propagation to topic and subscription confirmation. Still, Hedera's lightweight, append-only topic model avoids the need for smart contract execution or state-heavy ledger modifications, thereby enabling faster and more scalable identity operations.

In summary, the performance differences observed in the DID Update, Revoke, and Delete operations align with the trade-offs inherent to each platform's design. Ethereum's contract-centric architecture offers expressiveness at the cost of latency and volatility. XRPL achieves faster and more stable performance through a transaction-based model with deterministic finality. Hedera delivers the best latency characteristics overall, enabled by its event-based messaging and efficient consensus pipeline, albeit with minor variance due to SDK and network-layer propagation steps.

D. Privacy Analysis

1) Metadata-Leakage Score (MLS): MLS quantifies the amount of identifying or fingerprintable information revealed through the on-chain metadata of DID operations. Conceptually, MLS calculates the Shannon entropy of observable transaction payloads, thus assessing the diversity and uniqueness of metadata content. Expressed in bits per operation, higher MLS values indicate increased information leakage and greater privacy risk, whereas lower values imply stronger resistance to fingerprinting.

MLS analysis considers a predefined set of on-chain attributes available to any observer (e.g., blockchain indexers or mirror nodes) when a DID operation is performed. These attributes include DID-specific fields—such as decentralized identifiers, DID Document parameters (service endpoints, keys), and cryptographic signatures—as well as blockchain-specific metadata such as sequence numbers, block numbers, transaction hashes, and consensus timestamps. Off-chain data, such as client IP addresses, TLS metadata, or local timestamps, are explicitly excluded from consideration.

In this study, MLS is evaluated exclusively for three on-chain DID operations: Update, Revoke, and Delete, on Ethereum, Hedera, and XRPL. The analysis uses the same set of 100 transactions per operation previously employed for latency and cost benchmarking, ensuring methodological consistency. Payload metadata from these transactions is aggregated to compute MLS. An example of the metadata payload from a single DID Update operation (transaction) on Hedera is provided below:

```
{
  "transaction_info": {
    "initial_transaction_id": {
      "account_id": "0.0.5436919",
      "nonce": 0,
      "scheduled": false,
      "transaction_valid_start": "1747825185.015271329"
    },
    "number": 1,
    "total": 1
  },
  "consensus_timestamp": "1747825198.342924000",
  "message": {
    "message": {
      "timestamp": "2025-05-21T10:59:57.905Z",
      "operation": "update",
      "did": "did:hedera:testnet:zEqgDR8Fh2ZXbuLQTi1Bc_0.0.5649399",
      "event": {
        "Service": {
          "id": "did:hedera:testnet:zEqgDR8Fh2ZXbuLQTi1Bc_0.0.5649399#service-0",
          "type": "LinkedDomains",
          "serviceEndpoint": "https://example.com/0"
        }
      }
    },
    "signature": "hdcVRQRQgWHs1B8SDHhTebId0YilcB/YThF3Y18fuwmbxcvMaJTzMzjMucKJl5=="
  },
  "payer_account_id": "0.0.5436919",
  "running_hash": "kDGNsUl2nWzCZ9eOMGWT1tUvILBE2xpucpsWu6EtK6hzCXOxOMGxoahpxIi6EhvY",
  "running_hash_version": 3,
  "sequence_number": 818,
  "topic_id": "0.0.5649399"
}
```

The MLS calculation proceeds in two main stages:

1) **Tokenization and Entropy Calculation:** All decoded metadata payloads from the 100 transactions for each operation and platform are concatenated and tokenized into discrete units. Practically, this involves parsing each payload's JSON structure, flattening nested objects into key-value pairs, and treating each pair as an individual token. The empirical probability mass function $p(x)$ for each distinct token x is estimated. Shannon entropy per token is calculated as:

$$H_{\text{token}} = -\sum_{x} p(x) \log_2 p(x) \quad (1)$$

and then normalized by the total token count, yielding average entropy per token.

2) **Aggregation:** The entropy per token is scaled by the average token count per DID operation to produce the final MLS metric:

$$\text{MLS}_{\text{op}} = H_{\text{token}} \times \mathbb{E}[\text{tokens per transaction}] \quad (2)$$

Given the significant variability in payload structure and content length across different blockchains and DID operations, this scaling ensures fair comparisons of metadata leakage per operation. An MLS of zero bits per operation would indicate completely uniform payloads, indistinguishable across all observed transactions. Conversely, higher MLS values reflect sufficient variability—either in structure or content—to uniquely identify or fingerprint individual transactions.

Employing Shannon entropy to measure metadata leakage aligns with established practices in privacy and fingerprinting research. For instance, [32] applied entropy analysis to browser configuration samples, reporting that typical browser fingerprints exposed approximately 18.1 bits of identifying information. Similarly, [33] introduced entropy-based metrics for anonymity evaluation in mix networks, demonstrating that higher entropy corresponded to greater anonymity. Building upon these methodologies, our MLS adapts this information-theoretic approach specifically for evaluating privacy implications of on-chain DID metadata.

TABLE VI: MLS for DID Operations

(a) DID Update transactions

Chain	Bits/Token	Avg Tokens	Bits/Txn
Ethereum	0.0034	24.0	0.082
Hedera	0.0037	**19.0**	**0.071**
XRPL	**0.0024**	36.0	0.088

(b) DID Revoke transactions

Chain	Bits/Token	Avg Tokens	Bits/Txn
Ethereum	0.0036	23.0	0.083
Hedera	0.0040	**17.0**	**0.068**
XRPL	**0.0024**	35.0	0.083

(c) DID Delete transactions

Chain	Bits/Token	Avg Tokens	Bits/Txn
Ethereum	0.0035	22.0	0.078
Hedera	0.0038	**17.0**	**0.064**
XRPL	**0.0020**	40.0	0.079

2) Operation-Specific Leakage: Table VI summarizes the MLS results for each on-chain DID operation across the evaluated DLT platforms. Three key metrics are reported per platform and operation: bits per token, average tokens per transaction, and total bits per transaction (operation).

Across all evaluated operations, Hedera consistently demonstrates the highest bits-per-token values, indicating greater variability and uniqueness in its individual metadata fields. Ethereum follows closely behind, showing slightly lower per-token entropy, while XRPL records the lowest bits-per-token, reflecting a more uniform and compact metadata structure.

Regarding average tokens per transaction, XRPL generates the most verbose payloads, with approximately 35 to 40 tokens per transaction. This higher token count significantly amplifies its overall information leakage, despite each individual token having relatively low entropy. Ethereum occupies a moderate position, with payloads comprising around 22–24 tokens per transaction, and Hedera exhibits the leanest metadata footprint, averaging only 17–19 tokens per transaction.

When combining these two factors into total bits per transaction, XRPL's high token count results in the greatest leakage for the Update (0.088 bits/txn) and Delete (0.079 bits/txn) operations, closely matching Ethereum for Revoke operations (0.083 bits/txn). Conversely, Hedera consistently shows the lowest total leakage across all three operations—0.070 bits/txn for Update, 0.068 bits/txn for Revoke, and 0.064 bits/txn for Delete.

3) Aggregate Leakage Across All Operations: Table VII and Figure 10 summarize the aggregate entropy for each blockchain, along with the average bits per token and average tokens per transaction, combining results from all three on-chain operations.

An MLS value of exactly 0 bits per transaction would imply complete indistinguishability among on-chain DID operations. In practice, observed MLS values range between 0.06–0.09 bits per transaction for individual operations, scaling to 0.20–0.25 bits per transaction when aggregated across Update, Revoke, and Delete. These non-zero scores indicate measurable entropy—reflecting the presence of identifiable variability—in every on-chain DID payload.

Among the platforms tested, Hedera achieves the lowest aggregate MLS (0.20 bits per transaction, 60 bits total across all 300 transactions), benefiting from its comparatively compact payload structure. Ethereum has an intermediate MLS value of 0.24 bits per transaction, translating to a total of 72 bits. XRPL exhibits the highest overall MLS at 0.25 bits per transaction (75 bits in total), primarily driven by its larger token counts per transaction, despite having the lowest per-token entropy. The total bits metric highlights these cumulative differences, providing a clearer measure of overall metadata leakage across the evaluated operations.

Figure 10 further illustrates that within each blockchain, Delete operations leak the least metadata, followed by Update, and finally Revoke operations, although differences between operations remain modest.

Given that MLS correlates directly with the potential for transaction fingerprinting or correlation by external observers, Hedera's lower MLS suggests it offers a slightly stronger privacy posture for DID operations. XRPL's larger, more verbose payloads incur greater metadata leakage risk, while Ethereum positions itself between the two. Although these MLS differences appear minor when considered individually, their cumulative effect could influence the ease with which external observers or adversaries track or correlate DID-related activities over time.

TABLE VII: Total MLS across all operations

Chain	AvgBits/Token	AvgTokens/Txn	TotalBits/Txn	Total Bits
Ethereum	0.0035	23	0.24	72
Hedera	0.0038	**17.7**	**0.20**	**60**
XRPL	**0.0023**	37	0.25	75

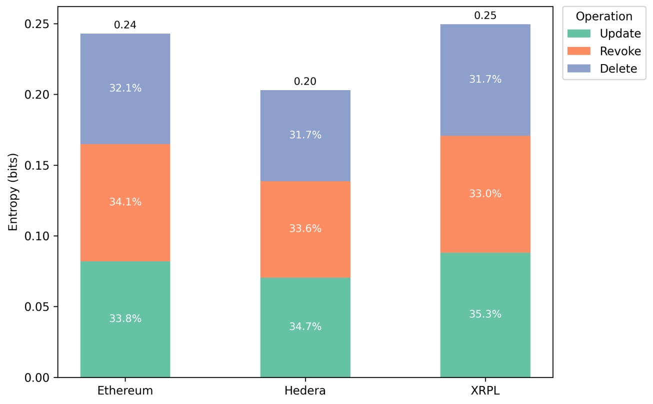


Fig. 10: Total MLS: total entropy by operation and chain.

E. Implications for Practitioners and Researchers

The findings of this study offer actionable guidance for practitioners selecting DID systems for deployment. In addressing RQ3, we identify clear trade-offs among latency, cost, and privacy, showing that no single platform optimally satisfies all dimensions—highlighting the importance of aligning platform choice with application-specific requirements.

Practitioners prioritizing low latency will find Hedera the most responsive. Its event-driven design yields the fastest DID operation times (4.0–4.4 seconds), though occasional outliers arise from asynchronous message propagation. Hedera also demonstrates the lowest metadata leakage, making it favorable for privacy-conscious deployments.

XRPL delivers highly predictable performance with minimal fee volatility. Its deterministic consensus mechanism ensures consistent latencies (5.6–5.8 seconds) and uniform transaction costs as low as $0.000021, making it well-suited for large-scale, cost-sensitive applications. However, its verbose transaction payloads result in the highest MLS, which may be a concern in privacy-critical contexts.

Ethereum, while exhibiting the highest costs and longest latencies (12.2–12.9 seconds for on-chain operations), maintains consistent performance tied to block time and benefits from mature tooling and deep Web3 integration—valuable in ecosystems requiring programmability and interoperability.

These platform-specific profiles support informed deployment decisions: Hedera excels in latency-sensitive scenarios, XRPL offers cost efficiency at scale, and Ethereum prioritizes integration flexibility. For researchers, our contributions include a reproducible benchmarking framework, a DID system evaluation model, and the first application of entropy-based privacy metrics to DID operations—laying groundwork for future empirical studies in decentralized identity.

F. Threats to Validity

While the benchmarking framework was designed for fairness and repeatability, several limitations must be acknowledged. First, all experiments were conducted under controlled testnet conditions, which may not fully reflect mainnet dynamics such as congestion, fee volatility, or security risk—posing a threat to external validity. Second, SDK behavior and performance may vary across versions or implementations; although we used widely adopted JavaScript SDKs, some platform-specific optimizations or limitations may affect generalizability. Finally, although we collected over 100 samples per operation, we did not perform statistical inference, and future work may include significance testing to support stronger conclusions. These factors should be considered when interpreting and applying the results.

VI. Interoperability in DID Systems

Interoperability in the context of decentralized identity systems refers to the ability of different DID methods and platforms to understand, resolve, and utilize each other's identifiers and credentials seamlessly. In practice, this requires adherence to common standards—such as the W3C's data model for DIDs—so that a DID created on one distributed ledger can be recognized and verified on another. Moreover, an interoperable DID system enables global resolvability, meaning that any DID can be resolved to its corresponding DID Document and public keys by any standards-compliant system. It also ensures that credentials issued to one DID can be trusted and verified across platforms.

A central tool in enabling such interoperability is the use of global resolvers. For instance, the `Universal Resolver (UR)` [34], an open-source project maintained by the DIF [35], serves as a meta-resolver that supports multiple DID methods via pluggable drivers. By querying the appropriate driver, the UR can retrieve a DID Document from a wide range of DLT-based and non-DLT-based methods, thereby standardizing the resolution process for developers and applications. Consequently, DID methods that provide an operational UR driver can be considered more interoperable in real-world deployments.

A. Comparative Analysis of Interoperability

Ethereum (did:ethr): The Ethr-DID method, built on Ethereum and formalized under ERC-1056, is one of the earliest W3C-conformant DID methods [36]. It is supported in the UR via an official driver and integrated into various identity frameworks, such as Veramo. Ethr-DID supports VC formats using both JWT and JSON-LD. Ethereum's identity ecosystem—including projects like Veramo and ConsenSys—has been actively involved in the W3C and DIF working groups, ensuring that Ethr-DID aligns with broader standards and can interoperate with other identity systems.

Hedera (did:hedera): Hedera joined the W3C and registered its DID method in the W3C DID Methods Registry in 2020 [37], signaling its commitment to emerging global standards. Hedera provides official SDKs and has integrated with SSI frameworks such as Hyperledger Aries. For example, a recent plugin enables Hedera DIDs and revocation registries to be used by Aries agents, demonstrating cross-platform identity integration [38]. Governance of Hedera's identity stack is managed by the Hedera Governing Council and the HBAR Foundation, both of which collaborate with standards bodies. Hedera is a member of both the DIF and the Linux Foundation's (LF) Decentralized Trust initiative and contributes actively to open-source identity projects, indicating a strong organizational commitment to interoperability. The entire Hedera codebase, including identity components, has been open-sourced under the LF's `Hiero` project, further aligning its governance with the open standards community. A UR driver is in development but is not yet fully integrated into the public resolver instance as of mid-2025.

XRPL (did:xrpl): The XRP Ledger introduced a native DID standard via the XLS-40d amendment in 2023, bringing decentralized identity capabilities to the XRPL. This DID method was explicitly designed to conform to the W3C DID v1.0 specification [39]. For DID resolution, XRPL's architecture supports global interoperability: any W3C-compliant resolver (e.g., the UR with an appropriate driver) can resolve `did:xrpl` identifiers by querying the ledger for the corresponding DID object and retrieving the linked DID Document. Organizationally, the XRPL DID effort has been led by RippleX (Ripple's development division) and the broader open-source XRPL community. Although Ripple and the XRPL Foundation were not initially members of the DIF or W3C working groups, the XLS-40d proposal was developed in alignment with W3C and decentralized identity infrastructure standards. Additionally, Ripple is rolling out a "Credentials" feature on XRPL that builds on the DID standard to support on-ledger verifiable credentials for KYC/AML compliance, following the W3C VCs format [40]. This demonstrates XRPL's focus on interoperable, privacy-preserving identity for regulated industries.

In summary, Ethereum, Hedera, and XRPL each contribute to a converging ecosystem of decentralized identity grounded in shared standards. All three evaluated DID methods demonstrate strong alignment with the W3C DID Core specification, establishing a baseline for technical interoperability. As shown in Table VIII, although these platforms follow a similar conceptual model of identity, they differ significantly in implementation approaches, governance structures, and deployment maturity. Ethereum's `did:ethr` method is widely adopted, benefits from mature tooling, and is fully integrated with the UR, making it highly interoperable in practice. Hedera's architecture supports scalable, event-driven identity management with partial UR integration and ongoing adoption within Aries-

TABLE VIII: Interoperability Comparison of DID Methods

Factor	Ethereum (Ethr-DID)	Hedera (HCS-DID)	XRPL (XLS-40d)
W3C DID Core Compliance	Yes	Yes	Yes
Standards Participation	Active (Veramo/Consensys) DIF member; W3C DID WG contributor	DIF and W3C member; contributor to Hiero	Aligns with W3C; not a DIF member
Universal Resolver Support	Yes (driver integrated)	Partial (driver under development)	Partial (no operational driver yet)
VC Format Support	W3C Verifiable Credentials (JSON-LD or JWT)	W3C Verifiable Credentials (JSON-LD); Hyperledger AnonCreds integration (via Aries plugin) for ZKP credentials	W3C Verifiable Credentials (JSON-LD); XRPL "Credentials" feature in progress
Governance Model	Open Ethereum network governance; DID method managed by open-source contract (ERC-1056), no central authority beyond Ethereum consensus	Public permissioned network (Hedera Council governance); identity features developed by Hedera & partners, now open-sourced under LF governance	Public ledger (validator governance via amendments); identity features proposed by RippleX and approved by validator consensus; Community-driven standardization
Cross-Chain Integration Potential	Multi-chain EVM support (mainnet, L2s, sidechains)	Interoperable via standards – e.g. Hedera DIDs used in Hyperledger Aries SSI networks	Early integration (e.g., Sologenic); cross-chain bridging under exploration

based SSI frameworks. XRPL's `did:xrpl` method, while relatively new, is fully W3C-compliant and tailored toward regulated financial applications, though it currently lacks full UR and cross-chain support.

Looking ahead, sustained collaboration through initiatives like the W3C, DIF, and open-source interoperability projects will be essential to advancing the interoperability landscape. This includes expanding universal resolver support for all DID methods and enabling seamless verification of credentials across ecosystems—such as a VC signed by a `did:ethr`, `did:hedera`, or `did:xrpl` being verifiable on any other compliant platform. Overall, while the DID methods differ in architectural design and governance, they share a common vision: enabling decentralized, portable digital identities. Each platform continues to evolve toward greater cross-ledger compatibility, and the growing convergence around W3C standards and cross-chain tooling suggests a future where user identities are no longer siloed to a single blockchain, but can instead move fluidly across a decentralized web of trust.

VII. Conclusion

This paper presented a comprehensive comparative analysis of DID systems across three prominent DLT platforms: Ethereum, Hedera, and the XRPL. Through systematic benchmarking of reference implementations—Ethr-DID, HCS-DID, and XLS-40d—we evaluated critical operational metrics, specifically latency, transaction costs, and metadata leakage, and interpreted the architectural design choices influencing real-world performance.

Our findings highlight substantial differences among the evaluated platforms. Ethereum, despite showing the highest absolute latencies and transaction costs, displays a tight correlation between DID operation performance and the network's inherent block production interval. This behavior underscores efficient integration with smart contracts and minimal overhead at the SDK level. Conversely, XRPL provides highly consistent latency and predictable cost profiles, positioning itself as a reliable and cost-effective platform for DID operations, albeit with slightly increased latency relative to its inherent consensus speed. Hedera emerges as the leader in terms of absolute latency, delivering rapid and scalable on-chain DID operations coupled with stable, minimal transaction fees.

The architectural analysis provides further insights into these empirical results. Ethereum's smart contract-centric model facilitates flexible identity metadata representation but introduces variability in transaction confirmation times due to gas-based prioritization mechanisms and network congestion. XRPL's lightweight ledger model offers deterministic behavior and stable transaction processing, contributing to its consistent performance. Hedera's event-driven, topic-based architecture supports high throughput, rapid resolution, and efficient identity workflows, with only minor variability arising from asynchronous event propagation.

Beyond empirical measurements, this study illustrates how foundational design decisions—including off-chain resolution strategies, ledger data structures, and transaction processing pipelines—directly affect the responsiveness and scalability of decentralized identity frameworks. These insights are invaluable for developers, enterprises, and ecosystem stakeholders tasked with selecting appropriate DID methods and underlying DLT platforms for deployment in real-world scenarios.

Taken together, our results point to a simple conclusion: the most effective design for a DLT-native DID system is an event-stream architecture with fast finality and deterministic low fees. This design combines the fast, ordered propagation we observed on Hedera with the predictable fees and stable behavior seen on XRPL, while also incorporating Ethereum's clean event semantics and mature account/key management patterns (delegates, multisig, key rotation) where needed. In practice, we recommend recording constant-size hash commitments on-chain, keeping encrypted, content-addressed DID documents off-chain, and representing each DID as an append-only log that supports ordered updates, key rotation, and recovery. This approach delivers low, stable latency and cost, minimizes metadata leakage, and keeps resolution straightforward—consistent with the patterns in our results.

Future research could extend these findings by evaluating additional DID methods across diverse blockchain networks, assessing performance under high-load conditions, and exploring the security trade-offs inherent in various decentralized identity architectures.

VIII. Appendix

A. Replication Package and Data Availability

To support transparency and reproducibility, we provide a complete replication package accompanying this paper. It includes the benchmarking scripts, raw result files, analysis notebooks, and documentation required to reproduce all figures and tables. The package is publicly available at: https://github.com/dlt-science/DID-Research. All materials are released under an open-source license and are compatible with the experimental methodology described in Section IV.

B. Summary Statistics for DID Operation Cost and Latency

TABLE IX: Summary statistics of DID operation latencies (in seconds) by DLT.

	Metric	**DID Create**	**DID Resolve**	**DID Update**	**DID Revoke**	**DID Delete**	**Full Cycle**
Ethereum	Mean	0.011	0.534	12.885	12.232	12.567	38.230
	Std	0.005	0.063	4.931	3.488	4.815	7.387
	Min	0.010	0.423	8.670	8.685	4.628	30.701
	Max	0.015	0.966	37.643	37.280	37.474	63.729
XRPL	Mean	5.602	0.076	5.821	5.761	5.683	22.943
	Std	1.183	0.006	1.197	1.199	1.081	2.221
	Min	2.594	0.075	3729	3708	3743	16183
	Max	7.229	0.078	8.347	8.356	7.342	27.739
Hedera	Mean	4.375	0.056	4.199	3.970	3.999	16.599
	Std	2.396	0.008	1.504	0.537	0.441	2.820
	Min	2.570	0.051	2.698	2.225	2.875	13.091
	Max	23.192	0.100	14.128	5.328	5.231	34.932

TABLE X: Summary statistics of DID operation costs (in USD) by DLT.

	Metric	**DID Create**	**DID Resolve**	**DID Update**	**DID Revoke**	**DID Delete**	**Full Cycle**
Ethereum	Mean	0.000000	0.000000	0.065600	0.064800	0.060300	0.190700
	Std	0.000000	0.000000	0.004989	0.005021	0.005214	0.008791
	Min	0.000000	0.000000	0.060000	0.060000	0.050000	0.170000
	Max	0.000000	0.000000	0.070000	0.070000	0.070000	0.210000
XRPL	Mean	0.000021	0.000000	0.000021	0.000021	0.000021	0.000084
	Std	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
	Min	0.000021	0.000000	0.000021	0.000021	0.000021	0.000084
	Max	0.000021	0.000000	0.000021	0.000021	0.000021	0.000084
Hedera	Mean	0.000159	0.000000	0.000156	0.000153	0.000150	0.000618
	Std	0.000000	0.000000	0.000001	0.000001	0.000000	0.000001
	Min	0.000159	0.000000	0.000156	0.000152	0.000150	0.000617
	Max	0.000159	0.000000	0.000156	0.000154	0.000150	0.000619

References

[1] A. Preukschat and D. Reed, *Self-Sovereign Identity: Decentralized digital identity and verifiable credentials*. Manning Publications, 2021.

[2] A. A. Malik, H. Anwar, and M. A. Shibli, "Federated Identity Management (FIM): Challenges and opportunities," in *Proceedings - 2015 Conference on Information Assurance and Cyber Security, CIACS 2015*, pp. 75–82, Institute of Electrical and Electronics Engineers Inc., 2016.

[3] A. Goel and Y. Rahulamathavan, "A Comparative Survey of Centralised and Decentralised Identity Management Systems: Analysing Scalability, Security, and Feasibility," 2025.

[4] A. Satybaldy, A. Subedi, and S. M. Idrees, "Decentralized key management for digital identity wallets," in *Blockchain Transformations: Navigating the Decentralized Protocols Era*, pp. 47–58, Springer, 2024.

[5] W3C, "Decentralized Identifiers (DIDs) v1.0," tech. rep., W3C Recommendation, 2022. https://www.w3.org/TR/did-1.0/.

[6] M. Sporny, D. Longley, D. Chadwick, and I. Herman, "Verifiable Credentials Data Model v2.0," tech. rep., W3C Candidate Recommendation Draft, 2025.

[7] Decentralized Identity Foundation, "DID-JWT," 2025. https://github.com/decentralized-identity/did-jwt.

[8] N. Helmy, "JWT vs Linked Data Proofs: comparing VC assertion formats," 2020. https://medium.com/mattr-global/jwt-vs-linked-data-proofs-comparing-vc-assertion-formats-a2a4e6671d57.

[9] P. Dunphy and F. A. Petitcolas, "A first look at identity management schemes on the blockchain," *IEEE Security & Privacy*, vol. 16, no. 4, pp. 20–29, 2018.

[10] A. Mühle, A. Grüner, T. Gayvoronskaya, and C. Meinel, "A survey on essential components of a self-sovereign identity," *Computer Science Review*, vol. 30, pp. 80–86, 2018.

[11] E. Krul, H.-y. Paik, S. Ruj, and S. S. Kanhere, "Sok: Trusting self-sovereign identity," *arXiv preprint arXiv:2404.06729*, 2024.

[12] A. Satybaldy, M. Nowostawski, and J. Ellingsen, "Self-sovereign identity systems: Evaluation framework," in *Privacy and Identity Management. Data for Better Living: AI and Privacy: 14th IFIP WG 9.2, 9.6/11.7, 11.6/SIG 9.2. 2 International Summer School, Windisch, Switzerland, August 19–23, 2019, Revised Selected Papers 14*, pp. 447–461, Springer, 2020.

[13] O. Dib and H. Toumi, "Blockchain for decentralized identity management: A comprehensive survey," *Digital Communications and Networks*, vol. 6, no. 3, pp. 256–271, 2020.

[14] L. Amherd, S.-N. Li, and C. J. Tessone, "Centralised or decentralised? data analysis of transaction network of hedera hashgraph," 2023.

[15] Y. Ucbas, A. Eleyan, M. Hammoudeh, and M. Alohaly, "Performance and scalability analysis of ethereum and hyperledger fabric," *IEEE Access*, vol. 11, pp. 67156–67167, 2023.

[16] M. Touloupou, K. Christodoulou, and M. Themistocleous, "Validating the blockchain benchmarking framework through controlled deployments of xrpl and ethereum," *IEEE Access*, vol. 12, pp. 22264–22277, 2024.

[17] Q. Noor, A. Multazam, S. G. Riyadi, R. Pratama, and A. Maulana, "Large-scale ethereum network performance analysis: Case study and implications," *Journal of Advanced Technological Innovations*, vol. 1, no. 1, pp. 26–34, 2024.

[18] A. Satybaldy, M. S. Ferdous, and M. Nowostawski, "A Taxonomy of Challenges for Self-Sovereign Identity Systems," *IEEE Access*, vol. 12, pp. 16151–16177, 2024.

[19] Hedera Hashgraph, "Hedera DID Method Specification," tech. rep., 2024. https://github.com/hashgraph/did-method/blob/master/hedera-did-method-specification.md.

[20] Hedera Hashgraph, "Hedera DID Javascript SDK ," 2025. https://github.com/hashgraph/did-sdk-js.

[21] Decentralized Identity Foundation, "Ethereum DID Method Specification," 2022. https://github.com/decentralized-identity/ethr-did-resolver/blob/master/doc/did-method-spec.md.

[22] Veramo, "Ethr-DID Library ," 2021. https://www.npmjs.com/package/ethr-did/v/2.1.2.

[23] Veramo, "Performant and modular APIs for Verifiable Data and SSI," 2025. https://veramo.io/, Accessed July 2025.

[24] XRPLF, "0040 XLS-40d: Decentralized Identity on XRPL ," 2023. https://github.com/XRPLF/XRPL-Standards/discussions/100.

[25] A. Malhotra, "XRPL Decentralized Identifier (DID) Specification," 2023. https://github.com/XRPLF/XRPL-Standards/tree/master/XLS-0040-decentralized-identity.

[26] XRPL Foundation, "XRPL JS SDK," 2025. https://github.com/XRPLF/xrpl.js.

[27] Decentralized Identity Foundation, "Ethereum DID Resolver," 2025. https://github.com/decentralized-identity/ethr-did-resolver/.

[28] Investopedia, "What Is Block Time? What It Measures, Verification, and Example," 2023. https://www.investopedia.com/terms/b/block-time-cryptocurrency.asp, Accessed June 2025.

[29] CryptoQuant, "Metrics: Mean Block Interval," 2025. https://cryptoquant.com/asset/xrp/chart/network-stats/block-interval.

[30] XRPL, "Transaction Cost," 2025. https://xrpl.org/docs/concepts/transactions/transaction-cost.

[31] Hedera, "Hedera Network Fees," 2025. https://docs.hedera.com/hedera/networks/mainnet/fees.

[32] P. Eckersley, "How unique is your web browser?," in *International Symposium on Privacy Enhancing Technologies Symposium*, pp. 1–18, Springer, 2010.

[33] C. Diaz, S. Seys, J. Claessens, and B. Preneel, "Towards measuring anonymity," in *International Workshop on Privacy Enhancing Technologies*, pp. 54–68, Springer, 2002.

[34] DIF, "Universal Resolver," 2025. https://dev.uniresolver.io/, Accessed July 2025.

[35] DIF, “Decentralized Identity Foundation,” 2025. https://identity.foundation/, Accessed July 2025.

[36] Consensys, “ERC-1056: Ethereum Lightweight Identity,” 2018. https://eips.ethereum.org/EIPS/eip-1056, Accessed July 2025.

[37] H. Team, “Hedera Hashgraph Joins World Wide Web Consortium (W3C),” 2020. https://hedera.com/blog/, Accessed July 2025.

[38] L. D. Trust, “Building Trust with Code: How DSR, Hedera, and LF Decentralized Trust, Are Advancing Decentralized Identity,” 2025. https://www.lfdecentralizedtrust.org/blog/, Accessed July 2025.

[39] Sologenic, “Decentralized Identity on the XRPL: Unpacking the New XLS-40 Proposal,” 2023. https://sologenic.medium.com/decentralized-identity-on-the-xrpl-unpacking-the-new-xls-40-proposal-2262a5972f20, Accessed July 2025.

[40] J. Maldonado, “Ripple to Launch “Credentials,” a Feature to Boost Institutional Adoption of XRPL,” 2025. https://news.bit2me.com/en/ripple-lanzara-credentials-para-adopcion-institucional-de-xrpl, Accessed July 2025.