Establishing a Baseline for Evaluating Blockchain-Based Self-Sovereign Identity Systems: A Systematic Approach to Assess Capability, Compatibility, and Interoperability

Wei Yao
New Jersey Institute of Technology
Newark, USA
wy95@njit.edu

Wenlu Du
New Jersey Institute of Technology
Newark, USA
wd48@njit.edu

Jingyi Gu
New Jersey Institute of Technology
Newark, USA
jg95@njit.edu

Junyi Ye
New Jersey Institute of Technology
Newark, USA
jy394@njit.edu

Fadi P. Deek
New Jersey Institute of Technology
Newark, USA
fadi.deek@njit.edu

Guiling Wang
New Jersey Institute of Technology
Newark, USA
gwang@njit.edu

ABSTRACT

Self-Sovereign Identity (SSI) is an evolving means of identity management that aims to give individuals more control over their digital identities and personal data rather than relying on third-party organizations or government authorities. Blockchain technology has the potential to strengthen SSI significantly by providing a secure and decentralized method towards managing and storing personal data, rendering them resistant to tampering and also enhancing their privacy. Given the diversity of blockchain-based SSI platforms, including Sovrin, uPort, and Hyperledger Indy, it is essential to have a consistent approach to evaluate the different SSI systems consistently. This paper offers guidelines for building systematic architecture and defining comprehensive internal interactions for a complete SSI system, thereby providing a framework for routine evaluation and analysis of an existing SSI platform and establishing base standards for assessing capability, compatibility and interoperability of SSI systems. Accordingly, the paper also reports on comprehensive experiments over many existing blockchain-based SSI systems using a multi-layered approach.

CCS CONCEPTS

• **Information systems → Information systems applications**; • **Software and its engineering → Software development techniques**.

KEYWORDS

Self-Sovereign Identity, Blockchain, Evaluation Framework, Test Coverage Criteria, Baseline

ACM Reference Format:
Wei Yao, Wenlu Du, Jingyi Gu, Junyi Ye, Fadi P. Deek, and Guiling Wang. 2024. Establishing a Baseline for Evaluating Blockchain-Based Self-Sovereign Identity Systems: A Systematic Approach to Assess Capability, Compatibility, and Interoperability. In *2024 6th Blockchain and Internet of Things Conference (BIOTC 2024), July 19–21, 2024, Fukuoka, Japan.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3688225.3688239

1 INTRODUCTION

The intentional, unauthorized use of a person's identifying information, known as identity theft, is a growing problem. In 2016, approximately 10% of U.S. adults experienced identity theft [10], and this number rose to 20% in 2021 due to the increased online shopping during the pandemic [11]. Therefore, a truly independent digital identity that no other entity (e.g., person, company, or government) can take away has been repeatedly sought in recent years, motivating the development of Self-Sovereign Identification (SSI) systems [45]. This shift to decentralized identity management, leveraging blockchain technology, constitutes a solution that eliminates the single point of failure, making it more difficult for hackers to steal identities. Compared with traditional identity management systems, where information on individuals is typically handled by third-party organizations or government authorities, SSI systems provide individuals greater control over their data and how it is utilized, thereby improving security and privacy. The implementation of the SSI approach can also have positive effects on enterprises and organizations. By granting customers greater control over their data, SSI allows businesses to increase customer trust without compromising data protection standards.

As a result, the Hyperledger Community [28] and Decentralized Identity Foundation [15] have launched an initiative to meet the needs of this emerging market, where the former is an open-source community for developing blockchain projects and a suite of stable frameworks for deployment, and the latter is an engineering-driven organization focused on the development of decentralized identity ecosystems. These efforts strengthen the work of researchers and organizations seeking to build SSI systems with full user identity control [43]. With more than 15 Hyperledger projects underway as of now, and the fast continuing proliferation of Hyperledger applications, frameworks, libraries, and decentralized identity ecosystems with different standards, operational complications are not unexpected. Given this, various issues may arise when multiple systems work together for the same purpose without a baseline standard. For instance, different systems might have varying data formats, protocols, and communication methods, leading to interoperability issues. Additionally, inconsistencies in how various systems handle certain tasks or processes can result in outcome discrepancies. Managing and maintaining the overarching system can become increasingly complex as new components or systems are added, leading to higher maintenance costs and potential technical debt [6]. Furthermore, the lack of a baseline standard can result in inefficiencies and waste, as different parties may independently develop similar solutions for the same problem, leading to duplication of efforts. Provided with so many options, it is becoming more challenging and time-consuming to build scalable and adaptive SSI systems. To the best of our knowledge, there is no consistent and reliable evaluation framework that can provide an objective assessment of SSI systems [42], and thus propose to build such an evaluation framework and offer here our own guidelines for the architecture and internal interactions of an SSI system. Employing a robust testing framework affords developers a secure, functional, and easy-to-use enterprise to identify and remedy bugs and vulnerabilities at an early stage and before the system is deployed. Our proposed framework follows the Foundational Principle of SSI [41]. It considers several important factors, including 1) the level of control that individuals have over their data, 2) the level of security provided by the system, 3) the ease of use for both individuals and organizations, 4) the integration with other systems, and 5) the scalability and flexibility of the system. Additionally, during the design of our test cases, we sought to identify the essential components and the critical workflows between these key components of SSI systems. And to demonstrate the effectiveness of the proposed evaluation framework, we performed comprehensive experiments on existing SSI systems. Given our design principles, we believe this evaluation framework will also perform well with newly deployed systems. With this being a first attempt at providing a baseline standard for evaluating SSI systems, the main contributions of the research presented in this paper are as follows: (1) A comprehensive introduction to the SSI framework, from architecture to workflows, covering the entire system. (2) A consistent and systematic methodology to evaluate SSI systems and their components. To do this, we assembled 45 test cases for blockchains, agents, and wallets to assess the capabilities of each component. Additionally, we introduced seven critical test suites, each consisting of a specific set of test cases designed to evaluate the compatibility and interoperability of the components within the SSI ecosystem. (3) A tangible instance of evaluating and analyzing existing commercial, academic, and open-source SSI systems with high adoption rates and notable reputations. Nine blockchains, 17 agents, and ten wallets were evaluated, showcasing the application of our evaluation methodology and further demonstrating its effectiveness in assessing SSI systems.

The rest of the paper is structured as follows: Section 2 highlights some important related work. Section 3 introduces the fundamentals of the SSI framework with its essential components and workflows. Section 4 presents the methodologies and building blocks for evaluating and analyzing existing SSI systems, including differential testing. Section 5 provides specific examples of the experimental environment, setup, and evaluation results, and Section 6 concludes with a summary of our research objectives.

2 RELATED WORK

There is a growing interest in using blockchain technology for identity management (IDM), specifically for Self-Sovereign identity (SSI) and Decentralized Identifier (DID) [5]. This interest stems from blockchains being immutable public ledgers that cannot be altered. A large body of literature offers a comparative analysis of blockchain-based IDM in academic and commercial areas [16, 17, 26]. Much of current studies pay particular attention to the three popular DID services developed recently: Sovrin, uPort, and Hyperledger Indy [43]. These services use blockchain technology to store and manage digital identities, providing individuals with increased control over their data and its utilization. Kondova and Erbguth [35] provided an overview of existing SSI on Hyperledger Indy public permissioned blockchain and uPort and Jolocom on the Ethereum public permissionless blockchain. Dunphy and Petitcolas [16] evaluated three representative services, specifically uPort, ShoCard, and Sovrin, by a seminal framework to characterize the nature of successful IDM schemes. In addition, several studies focus on more comprehensive efforts for comparing SSI framework and DID systems. Kim et al. [33] conducted a study on the security of DID systems. They investigated its components and analyzed their interactions, then identified security threats and vulnerabilities during the deployment. Schardong and Custódio [46] examined both conceptual and practical advances in SSI and proposed a novel taxonomy to categorize SSI research. Čučko et al. [14] provided a comprehensive collection and analysis of various SSI properties and presented a general SSI process with highlighted process steps with important individual properties during implementation. Zaeem et al. [43] presented a comprehensive list of functional requirements of SSI and their use cases in both commercial and academic areas.

Despite numerous studies, the current evaluation framework for SSI lacks comprehensive experimentation with consistent guidelines from the capability, compatibility and interoperability point of view. Thus, based on our review of [14, 16, 33, 43, 46], we choose these DLTs [8, 18, 23, 29, 30, 38, 40, 52, 55], agents [1–4, 9, 22, 24, 31, 32, 34, 39, 48, 50, 56, 58, 59, 63], and wallets [9, 12, 13, 34, 47–49, 51, 57, 62] for the systematic evaluation.

3 BUILDING BLOCKS

The scope of a complete SSI system includes data models, participants, and components, as shown in Figure 1. In this section, we introduce the foundational knowledge of how a SSI system is built starting from its architecture through its workflows.

3.1 Data Models

Decentralized Identifiers (DIDs) [61] and Verifiable Credentials (VCs) [60] are two essential data models of SSI systems. (1) **DID** is a globally unique, resolvable, and self-governing identifier used to refer to an entity, such as a person or an organization. DIDs are generated and managed by the entity they designate and are stored on a distributed, tamper-resistant ledger, such as a blockchain. Thus, it can be verified without centralized registries or third-party identity providers. Each DID can be represented by a URI style [7] (i.e., *did:method:identifier*) and linked to a DID subject (often an identity holder) and a DID document that details its subject, cryptographic public keys, and other authentication techniques used to establish

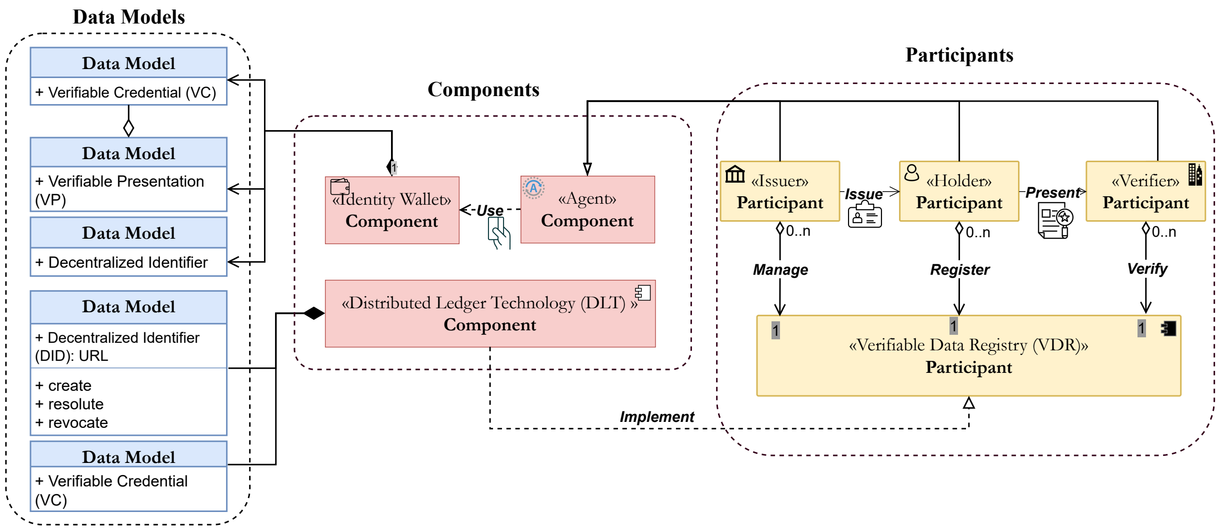

Data Models
Data Model
+ Verifiable Credential (VC)
Data Model
+ Verifiable Presentation (VP)
Data Model
+ Decentralized Identifier
Data Model
+ Decentralized Identifier (DID): URL
+ create
+ resolute
+ revocate
Data Model
+ Verifiable Credential (VC)
Components
«Identity Wallet» Component
Use
«Agent» Component
«Distributed Ledger Technology (DLT) » Component
Implement
Participants
«Issuer» Participant
Issue
«Holder» Participant
Present
«Verifier» Participant
0..n
Manage
Register
Verify
1
«Verifiable Data Registry (VDR)» Participant

Figure 1: Class diagram of SSI ecosystems

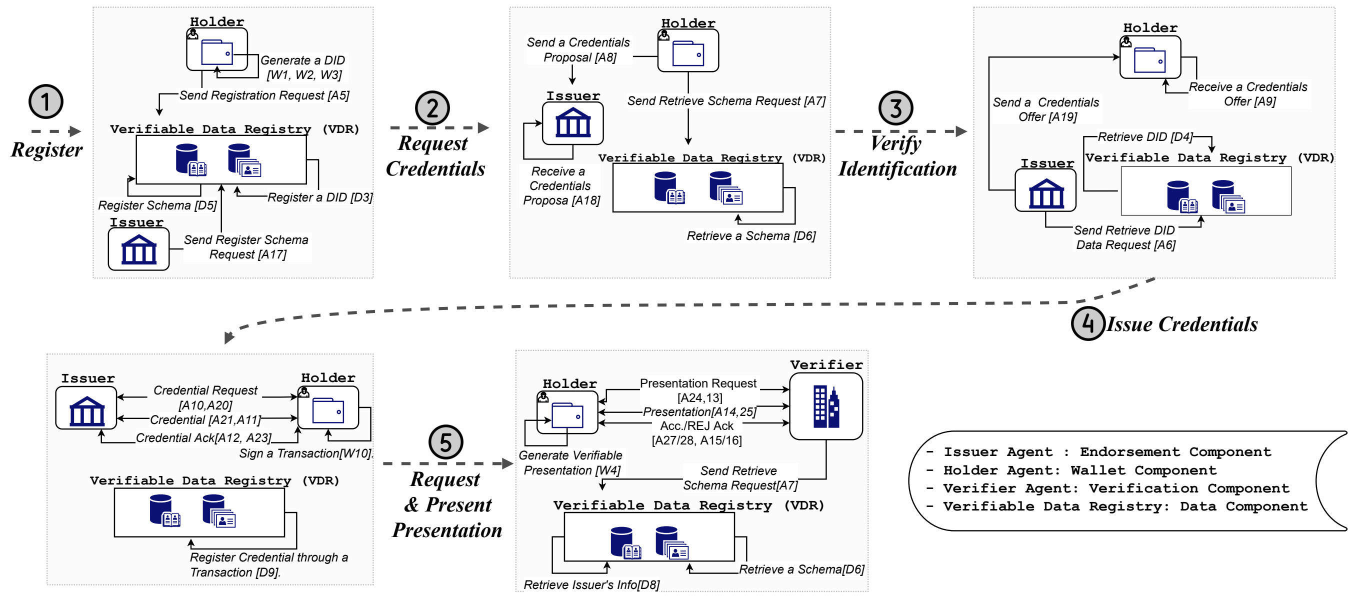
the holder's ownership. The DID method gives detailed specifics about the implementation method of the DID and the exact operations (creation, resolution, and revocation) of DIDs and DID documents. (2) **VCs** are digital representations of actual credentials, such as a driver's license, passport, or college degree, that include information about the subject (i.e., identification data and claim(s) describing the subject). VCs are issued by a reputable issuer, such as a government agency or an educational institution, cryptographically signed and kept on a distributed ledger, a blockchain for example, rendering them tamper-resistant and verifiable. To allow selective disclosure and preserve the VC holder's private information while in contact with other entities, cryptographic schema like Zero-Knowledge Proofs (ZKPs) [21, 36] can be used. VC holders may compile information from many VCs issued by multiple issuers into a single Verifiable Presentation (VP) [60], then share it with a particular verifier for confirmation.

3.2 Participants

In the context of SSI and DID, four critical participants are included. (1) **Issuer**: A reputable entity or organization that generates, signs and distributes credentials. The issuer is responsible for confirming the attributes of the holders' identities and issuing them verifiable credentials. (2) **Holder**: An entity or a person that possesses and controls credentials as digital identities. They own their digital identity and can control, distribute, and utilize their personal information from digital identities. (3) **Verifier**: An entity or organization requesting proof from a holder and verifying a holder's identity to provide access to a service or resource. In SSI, the verifier can verify the holder's identity without communicating with the issuer and storing any of the holder's information. (4) **Verifiable Data Registry (VDR)** [60]: Used to store, manage, distribute, and revoke DIDs, VCs, and other verifiable data in a secure and decentralized manner. In addition, a VDR facilitates the preservation of a VC's credentials schema to validate the VC. A VDR is self-sovereign, meaning that data owners have complete control over their data and can choose who has access to it. A VDR also gives transparency and accountability on how data is used and can provide a secure and verified method for sharing and accessing personal data, preventing fraud, enhancing industries' efficiency, and bridging the digital divide. A VDR can be implemented as distributed ledgers, decentralized file repositories, or any other distributed database. In the manner of SSI, blockchain as the distributed ledger is the key to implementing such registries.

3.3 Components

Implementing complete SSI systems, the following components are required. (1) **Distributed Ledger Technology (DLT)** [53], instead of a centralized database, DLT leverages blockchain to implement the VDR in SSI by providing trust, integrity, and availability. It works as a repository that stores the transactions used to record the event for credentials issuance from the issuer to the holder. In addition, some systems, like the Sovrin ledger, provide a universal DID resolver through the DLT, and store the verification information of a registered public DID's owner. (2) **Agent** is a standalone software program or service issuers, holders, or verifiers use. It can perform several functions, such as creating and managing DIDs, VCs, and related information, facilitating interactions with decentralized identity networks and services, protecting the privacy and security of individual data, and acting as an intermediary between an individual user and DLT. It provides additional functionality like key management, identity wallet backup/recovery, data storage, and DID communication. (3) **Identity Wallet** [44] acts as a digital wallet for a holder's digital identities and digital credentials, such as DIDs and VCs. It allows a holder to create, store, and manage their own digital identities and credentials and share them with other parties securely and verifiably. It also allows individuals to control access to their data and to have transparency and accountability for how it is used and shared. An identity wallet can be a software application that runs on a smartphone, tablet, or computer; or a hardware device such as a USB dongle or a smart card. It allows



1 Register
Holder
Generate a DID [W1, W2, W3]
Send Registration Request [A5]
Verifiable Data Registry (VDR)
Register Schema [D5]
Register a DID [D3]
Issuer
Send Register Schema Request [A17]
2 Request Credentials
Send a Credentials Proposal [A8]
Holder
Issuer
Send Retrieve Schema Request [A7]
Receive a Credentials Proposa [A18]
Verifiable Data Registry (VDR)
Retrieve a Schema [D6]
3 Verify Identification
Holder
Send a Credentials Offer [A19]
Receive a Credentials Offer [A9]
Retrieve DID [D4]
Issuer
Verifiable Data Registry (VDR)
Send Retrieve DID Data Request [A6]
4 Issue Credentials
Issuer
Holder
Credential Request [A10,A20]
Credential [A21,A11]
Credential Ack[A12, A23]
Sign a Transaction[W10].
Verifiable Data Registry (VDR)
Register Credential through a Transaction [D9].
5 Request & Present Presentation
Holder
Verifier
Presentation Request [A24,13]
Presentation[A14,25]
Acc./REJ Ack [A27/28, A15/16]
Generate Verifiable Presentation [W4]
Send Retrieve Schema Request[A7]
Verifiable Data Registry (VDR)
Retrieve Issuer's Info[D8]
Retrieve a Schema[D6]
- Issuer Agent : Endorsement Component
- Holder Agent: Wallet Component
- Verifier Agent: Verification Component
- Verifiable Data Registry: Data Component

Figure 2: Essential workflows of SSI

holders to control access to their data and have transparency and accountability for how it is used and shared. Typically, identity wallet utilizes blockchain, or other distributed ledger technologies, for secure storage and management of credentials. It can be deployed on issuer, holder, or verifier side. It generates private and public key pairs and securely stores the key pairs and received VCs.

4 METHODOLOGIES

To effectively evaluate Blockchain-based SSI systems, it is crucial to define and understand three key concepts: (1) **Capability**: In the context of Blockchain-based SSI systems, capability refers to the system's ability to perform its intended functions effectively. This includes the management of digital identities, issuing and verifing credentials, and performing cryptographic operations securely. (2) **Compatibility**: Refers to the degree to which instances of components can operate together with other such instances without any conflict in forming a complete SSI system. (3) **Interoperability**: The ability of different Blockchain-based SSI systems to work together and exchange information seamlessly.

To gain a comprehensive understanding of how to build a baseline standard for evaluating SSI systems' capability, compatibility, and interoperability, we investigated the following research questions (RQs). **RQ1: What are the key components within SSI systems, and how do they contribute to the overall functionality of the system?** This RQ aims to identify the essential components of SSI systems and the test cases utilized to assess each component's functionality and effectiveness. **RQ2: What are the critical workflows of SSI systems, and how do they evaluate the interactions between these key components?** The objective of this RQ is to pinpoint the crucial workflows between the key components of SSI systems and the test suites employed to evaluate capacity of tested components to form a complete SSI system. **RQ3: How can existing components in SSI be systematically compared and assessed using the proposed evaluation framework?** The goal of this RQ is to demonstrate the practical application of the proposed evaluation framework by comparing and analyzing existing components in SSI ecosystems, thereby showcasing its effectiveness in assessing various platforms. **RQ4: How can the applicability of the existing components in different layers of SSI systems be systematically compared and assessed using the proposed evaluation framework?** This RQ aims to demonstrate the practical experiment of the proposed evaluation framework by comparing and analyzing the interactions between the components in specific SSI ecosystems. **RQ5: To what extent do the proposed evaluation criteria and methodologies generalize across different SSI platforms, and how can they be adapted or extended to accommodate emerging trends and technologies in the SSI ecosystem?** The objective of this RQ is to illustrate how to utilize the proposed framework to evaluate the interoperability of existing SSI components in different SSI ecosystems.

By addressing these research questions, we aim to provide a comprehensive and systematic approach to evaluating SSI ecosystems. To answer RQ1 and RQ2, we first defined the scope of essential work within an SSI ecosystem. Figure 2 shows the essential workflows for a complete SSI ecosystem, each workflow following the **Foundational Principle of SSI** [41]. The following subsections present the corresponding test cases and test suites designed to assess an SSI ecosystem.

4.1 Test Cases Design

Answer to RQ1. In this study, functional scenarios are established based on the workflows proposed for the SSI architecture. Each scenario is then divided into several test cases, with the domain of input and expected output defined for each. A **Test**

Table 1: Essential Test Cases with Statements and Inputs: $D[x]$ for Blockchain, $A[x]$ for Agent, and $W[x]$ for Wallet

#	Statement	Input
D[1]	Register node with writing permission	node
D[2]	Retrieve node registration transaction	node ID
D[3]	Register DID through a transaction	DID
D[4]	Retrieve DID with data	DID
D[5]	Register schema through a transaction	schema
D[6]	Retrieve credentials schema	schema ID
D[7]	Register issuer's public verifiable information	issuer's pk
D[8]	Retrieve issuer's public verifiable information	issuer DID
D[9]	Register credentials through a transaction	credentials
D[10]	Retrieve transaction of issuing credentials	credentials
A[1]	Create connection invitation.	DID
A[2]	Accept another agent's invitation.	invitation
A[3]	Reject another agent's invitation.	invitation
A[4]	Establish connection between two agents.	invitation
A[5]	Send Register DID request.	DID
A[6]	Send Retrieve DID data request.	DID
A[7]	Send Retrieve Schema request.	schema ID
A[8]	Send credentials proposal.	proposal
A[9]	Receive credentials offer.	offer
A[10]	Send credentials request.	credentials ID
A[11]	Receive credentials.	credentials ID
A[12]	Send credentials acknowledgement.	ack
A[13]	Receive presentation request.	presentation
A[14]	Send presentation.	presentation
A[15]	Receive presentation acknowledgement.	ack
A[16]	Receive the rejection of presentation.	rejection
A[17]	Send Register Schema request.	schema
A[18]	Receive credentials proposal.	proposal
A[19]	Send credentials offer.	offer
A[20]	Receive credentials request.	credential ID
A[21]	Send credentials.	credentials
A[22]	Receive credentials acknowledgement.	ack
A[23]	Issue verifiable credentials.	DID
A[24]	Send presentation request.	presentation
A[25]	Receive presentation.	presentation
A[26]	Send presentation acknowledgement.	ack
A[27]	Reject presentation.	rejection
A[28]	Verify verifiable credentials.	credentials
W[1]	Generate master secret.	wallet
W[2]	Generate key pairwise.	wallet
W[3]	Generate DID.	sk, pk
W[4]	Generate Verifiable Presentation.	credentials
W[5]	Encrypted store private key.	sk
W[6]	Encrypted store DID.	DID
W[7]	Encrypted store credentials.	credentials
W[8]	Retrieve DID.	sk, pk
W[9]	Retrieve Verifiable Presentation.	DID
W[10]	Sign a trsanction by using private key.	txn,sk

Case (TC) is a minimum unit that cannot be split for a particular test. Let $I_x = x_1, x_2, ..., x_m$ and $O_y = y_1, y_2, ..., y_n$ be the set of input variables and output variables, and $V_z = v_1, v_2, ..., v_z$ be the set of values. Then the test case $\mathcal{TC}$ is defined as: $\mathcal{TC} = \{(x_1, v_{x1}), ..., (x_n, v_{xn}), (y_1, v_{y1}), ..., (y_m, v_{ym})\}$ A **Test Oracle** is a program that determines the expected output of a test case. It is used to verify whether a test case has passed or failed. For quantization, we use a boolean expression as an assertion to check the execution result of the test case. For example, a function for an addition operation should return the sum of its two inputs. The test oracle will determine if the returned value matches the expected outcome by placing an assertion. According to the functionalities of each component in the SSI framework, a list of test cases is compiled and proposed, which are shown in Table 1. The test oracle is also included to evaluate each test case. The essential test cases for each component of SSI ecosystems are designated as $D[x]$ for blockchain, $A[x]$ for agent, and $W[x]$ for wallet.

Table 2: Coverage and Path for Test Suites

#	Coverage and Path
$\mathcal{TS}_1$	(holder, W[1]), (holder, W[2]), (holder, W[3]), (holder, A[5]), (dlt, D[3])
$\mathcal{TS}_2$	(issuer, A[17]), (dlt, D[5])
$\mathcal{TS}_3$	(holder, A[7]) ,(dlt, D[6]), (holder, A[8]), (issuer, A[18])
$\mathcal{TS}_4$	(issuer, A[6]), (dlt, D[4]), (issuer, A[19]), (holder, A[9]), (holder, A[10]), (issuer, A[20]), (issuer, W[10]), (dlt, D[9]), (issuer, A[21]), (holder, A[11]), (holder, A[12]), (issuer, A[23])
$\mathcal{TS}_5$	(verifier, A[24]), (holder, A[13])
$\mathcal{TS}_6$	(holder, W[4]), (holder, A[14]), (verifier, A[25])
$\mathcal{TS}_7$	(verifier, A[7]), (dlt, D[6]), (dlt, D[8]), (verifier, A[28]), (dlt, D[10]), (verifier, A[26]/A[27]), (holder, A[15]/A[16])

4.2 Test Suites Design

Answer to RQ2. A complete SSI ecosystem should consist of the seven essential workflows shown in Figure 2. For each workflow, we use a test suite to complete the evaluation. A **Test Suites** is a collection of test cases combined in a sequential order to be executed for a particular testing purpose. If a test suite $\mathcal{TS}$ consists of $\mathcal{TC}_1, \mathcal{TC}_2, ..., \mathcal{TC}_w$, then the running result of the $\mathcal{TS}$ is $\mathcal{R}_1 \wedge \mathcal{R}_2 \wedge ... \wedge \mathcal{R}_w$. Table 2 shows the coverage and path of our test suites. The detailed evaluation algorithm for each test suite is shown below.

Register DID: This workflow serves as the foundation for the entire evaluation process. Participants can use their digital wallet to generate an asymmetric key-pair (public key denoted as pk and private key denoted as sk) and a DID. The DID is linked to the key-pair by using the public key as a unique identifier within the DID. Each participant must complete the DID registration process. Taking the holder as an example, the evaluation process can be described by the Algorithm 1.

Algorithm 1: Register DID Evaluation

```
Input: Holder $\mathcal{H}$, DLT dlt, TCs $\{W1, W2, W3, W3, A5, D3\}$
1 $(\mathrm{pk}^{\mathcal{H}}, \mathrm{sk}^{\mathcal{H}}) \leftarrow$ KeyGen()
2 if AssertFalse($(\mathrm{pk}^{\mathcal{H}}, \mathrm{sk}^{\mathcal{H}}), W2$) then return;
3 $\mathrm{DID}^{\mathcal{H}} \leftarrow$ DIDGen($sk^{\mathcal{H}}$)
4 if AssertFalse($\mathrm{DID}^{\mathcal{H}}, \mathcal{H}.W3$) then return;
5 if AssertFalse(request, $A5$) then return;
6 if AssertFalse(transaction, $D3$) then return;
7 AssertionTrue
```

Register Schema: The issuer generates the schema for credentials, including selected attributes and binding it with the issuer's public key for verification that the schema was created by this issuer. The schema is then registered to the DLT via a blockchain transaction. The evaluation can be represented as Algorithm 2.

Algorithm 2: Register Schema Evaluation

```
Input: Issuer $\mathcal{I}$, DLT dlt, TCs $\{A17, D5\}$
1 schema ← ScheGen($\{attributes\}$, $sk^{\mathcal{I}}$)
2 if AssertFalse(schema, $\mathcal{I}.A17$) then return;
3 def ← DefGen(schema, $pk^{\mathcal{I}}$)
4 if AssertFalse(def, dlt.$D5$) then return;
5 AssertionTrue
```

Request Credentials: After the credentials schema has been registered, the holder intends to obtain credentials based on the registered schema. Therefore, the holder sends a request containing the credentials parameters to the issuer. This process can be assessed using Algorithm 3.

Algorithm 3: Request Credentials Evaluation

```
Input: Holder $\mathcal{H}$, Issuer $\mathcal{I}$, DLT dlt, TCs $\{A7, D6, A8, A18\}$
1 if AssertFalse(schema, $\mathcal{H}.A7$) then return;
2 if AssertFalse(schema, dlt.$D6$) then return;
3 proposal ← ($DID^{\mathcal{H}}$, schema, $attributes^{\mathcal{H}}$)
4 if AssertFalse(proposal, $\mathcal{H}.A8$) then return;
5 if AssertFalse(proposal, $\mathcal{I}.A18$) then return;
6 AssertionTrue
```

Issue Credentials: During this process, the issuer initially verifies the correctness of the attribute values connected to the holder's DID, which the holder provided. Following verification, the issuer extends an offer to issue credentials to the holder. Once the holder accepts this offer, the issuer creates credentials for the holder by utilizing values from the holder's DID, generating the credentials based on the chosen schema and adding a signature. Subsequently, the issuer registers the credentials through a blockchain transaction. Upon completion of this step, a verifiable credential (VC) is generated, which can be fully cryptographically verified. After generating the VC, the issuer transmits it to the holder via a secure communication channel, like one established by their respective Agent applications. Lastly, the holder encrypts the verifiable credentials with a private key and stores them in their identity wallet. The evaluation can be assessed as Algorithm 4.

Request Presentation: If a verifier wishes to verify the holder's credentials (specifically, a particular attribute within the credentials) generated in the previous step, the verifier sends a presentation request to the holder, specifying the particular attributes that they seek to validate. This process can be assessed using Algorithm 5.

Present Presentation: Upon receiving the presentation request from the verifier, the holder checks their wallet to determine whether it contains a corresponding schema with the requested attributes. If such a schema exists, the holder selects credentials that include the requested attributes. The holder then generates a presentation of the credentials, which comprises: 1) metadata information related to the holder's credentials, 2) claims corresponding to the values of the requested attributes, and 3) a proof that serves as a cryptographically verifiable method of the credentials' issuer. Once the verifiable presentation has been generated, the holder sends it to the verifier through secure communication channels. The evaluation is illustrated in Algorithm 6.

Algorithm 4: Credential Issuance Evaluation

```
Input: Issuer $\mathcal{I}$, Holder $\mathcal{H}$, DLT dlt, TCs $\{A6, D4,$
       $A19, A9, A10, A20, W10, D9, A21, A21, A12, A23\}$
1 if AssertFalse($DID^{\mathcal{H}}$, $\mathcal{I}.A6$) then return;
2 if AssertFalse($DID^{\mathcal{H}}$, dlt.$D4$) then return;
3 if AssertFalse(offer, $\mathcal{I}.A19$) then return;
4 if AssertFalse(offer, $\mathcal{H}.A9$) then return;
5 if AssertFalse(Credentialrequest, $\mathcal{H}.A10$) then return;
6 if AssertFalse(Credentialrequest, $\mathcal{I}.A20$) then return;
7 Credential ← CredGen(Schema, $DID^{\mathcal{H}}$)
8 $\sigma_{Credential}$ ← Sign(Credential, $sk^{\mathcal{I}}$)
9 if AssertFalse($\sigma_{Credential}$, $\mathcal{I}.W10$) then return;
10 if AssertFalse(transaction, dlt.$D9$) then return;
11 VC ← VCGen(Credential, transaction)
12 if AssertFalse(VC, $\mathcal{I}.A21$) then return;
13 if AssertFalse(VC, $\mathcal{H}.A11$) then return;
14 if AssertFalse(VC.ACK, $\mathcal{H}.A12$) then return;
15 if AssertFalse(VC.ACK, $\mathcal{I}.A23$) then return;
16 AssertionTrue
```

Algorithm 5: Request Presentation Evaluation

```
Input: Verifier $\mathcal{V}$, Holder $\mathcal{H}$, TCs $\{A24, A13\}$
1 request ← Gen($\{attributes\}$)
2 if AssertFalse(request, $\mathcal{V}.A24$) then return;
3 if AssertFalse(request, $\mathcal{H}.A13$) then return;
4 AssertionTrue
```

Algorithm 6: Present Presentation Evaluation

```
Input: Holder $\mathcal{H}$, Verifier $\mathcal{V}$, TCs $\{W4, A14, A25\}$
1 if attribute $\notin$ VC then return;
2 if (Claim ← VC) and (Proof ← VC) then
3     Presentation ← $GenVP$(Metadata, Claim, Proof)
4 if AssertFalse(Presentation, $\mathcal{H}.W4$) then return;
5 if AssertFalse(Presentation, $\mathcal{H}.A14$) then return;
6 if AssertFalse(Presentation, $\mathcal{V}.A23$) then return;
7 AssertionTrue
```

Verify Credentials: Upon receiving the verifiable presentation from the holder, the verifier's primary task is to validate the authenticity, integrity, and correctness of the presented information. The verification process typically involves the following steps: 1) assessing the metadata information related to the holder's credentials to ensure the necessary attributes and schemas are included, 2) examining the claims provided for each requested attribute to ascertain their validity, and 3) checking the cryptographic proof that serves as a verifiable method of the credentials' issuer to confirm its legitimacy. Throughout the evaluation process, it is crucial for the verifier to ensure that the presentation was generated by the rightful holder, the information has not been tampered with, and the issuer's proof is valid. By systematically evaluating each step and adhering to established protocols and standards, the verifier can effectively assess the reliability of the received verifiable presentation. This process can be evaluated using Algorithm 7.

Algorithm 7: Verify Credentials Evaluation

```
Input: Verifier $\mathcal{V}$, DLT dlt, Holder $\mathcal{H}$, TCs
       $\{A7, D6, D8, A28, D10, A26/A27, A15/A16\}$
1  if AssertFalse(SchemaID, $\mathcal{V}.A7$) then return;
2  if AssertFalse(Schema, dlt.$D6$) then return;
3  if AssertFalse(DID$^{\mathcal{I}}$, dlt.$D8$) then return;
4  if AssertFalse(Presentation, $\mathcal{V}.A28$) then return;
5  Metadata, Claim, Proof ← Retrieve(Presentation)
6  if AssertFalse(Transaction, dlt.$D10$) then return;
7  Metadata$_{\text{dlt}}$ ← Retrieve(Transactions)
8  if Metadata $\notin$ Metadata$_{\text{dlt}}$ then return;
9  if (attribute $\in$ Claim) and ($\sigma_{\text{Credential}} \in$ Proof) then
10     if AssertFalse(Acceptance.Ack, $\mathcal{V}.A26$) then return;
11     if AssertFalse(Acceptance.Ack, $\mathcal{H}.A15$) then return;
12 else
13     if AssertFalse(Rejection.Ack, $\mathcal{V}.A27$) then return;
14     if AssertFalse(Rejection.Ack, $\mathcal{H}.A16$) then return;
15 AssertionTrue
```

5 EXPERIMENTS AND EVALUATION

Our evaluation consists of three components and was conducted on cloud servers hosted by Amazon Web Services (AWS).

5.1 Evaluation 1, Test Cases Coverage

Answer to RQ3. To provide a practical demonstration of our proposed evaluation framework, we have performed three groups of test cases. Each group targets existing, high-reputation SSI components from different layers of the SSI architecture. Consequently, the first group corresponds to DLTs, the second group to agents, and the third group to Wallets. For each group, we performed the test cases, as illustrated in Table 1, on each selected product.

In the first group of evaluation, the platforms we selected as DLTs are Azure Multi-chain, Bitcoin, Ethereum, FISCO-BCOS, Hyperledger Fabric, Hyperledger Indy, IOTA Tangle, Solana, and Sovrin Ledger. The evaluation results for selected DLTs are shown in Figure 3. Among all, Azure Multi-chain, FISCO-BCOS, Hyperledger Fabric, and Hyperledger Indy have passed all our test cases. We observed that the remaining ones entirely omit the process related to writing permission and authentication. The reason for these failed test cases is that the selected DLTs were originally built as public blockchains and used for different purposes rather than hybrid or consortium blockchains [64], which provide higher customization to support DLT for SSI systems. Bitcoin, for example, bypasses the retrieval and registration process of DID, as it was built much earlier than other DLTs and was only intended as a DLT for cryptocurrency. Researchers and developers can only leverage Bitcoin's original features rather than customize it to fit SSI. Overall, these limitations affect the implementation of an SSI ecosystem based on some existing DLTs that were built for other purposes.

In the second group, we selected the following platforms as agents: ACAPY, AF-GO, AF-JS, AF-NET, Blockcerts, Findy, Gateway, Indicio, JoJocom Agent, Microsoft Entra, SelfKey, Serto, TDID, Veramo, Verity, Walt.id SSI kit, and WeIdentity. The evaluation results for the selected agents are illustrated in Figure 4. ACAPY, Indicio, and Verity are the only three platforms that have successfully passed all the test cases. A common issue among the remaining agents was the lack of proper connection establishment before sending credentials. This limitation may potentially hinder secure and reliable communication between participants in the SSI ecosystem. As issuers, some of these agents failed to send and receive credential offers and proposals, which are critical steps in the credentials issuance process. On the other hand, as verifiers, some agents did not send presentation acknowledgments, an essential step for ensuring that the verification process has been completed successfully. These findings highlight the importance of thorough evaluation and testing of SSI agents to ensure they adhere to best practices and provide secure and reliable communication within the SSI ecosystem.

In the third group, we selected the following platforms as Wallets: Azure Key Vault, Blockcerts, Connect.Me, Cryptid, IndySDK, Jolocom Wallet, SelfKey, Selv, Trinsic, and Walt.id Wallet kit. The evaluation results for the chosen wallets are depicted in Figure 5. Azure Key Vault, Connect.Me, Trinsic, and Walt.id Wallet Kit passed all the test cases. We observed that half of the evaluated wallets were unable to generate and retrieve verifiable presentations, a crucial aspect of SSI functionality. This limitation may hinder the effective use of these wallets in real-world scenarios, as the generation and retrieval of verifiable presentations are essential steps in the credentials' verification process.

5.2 Evaluation 2, Test Suites Coverage

Answer to RQ4. To provide a practical demonstration of our proposed evaluation framework, we performed three groups of test suites to evaluate the capability of each component that serves in the same DLT as an autonomous domain.

To evaluate the interaction between components in different layers, we utilized the test suites proposed in Table 2. We calculated the path weight to assess the capability of selected components in different layers. For example, in test suite TS_2, to complete this test suite successfully, both $(issuer, A[17])$ and $(dlt, D[5])$ must be satisfied. If ACAPY is selected as the $issuer$ and Hyperledger Indy as the dlt, and both clauses yield a result of 1, then ACAPY and Hyperledger can complete test suite TS_2, and the path weight between ACAPY and Hyperledger Indy is incremented by one, denoted as $path(ACAPY, Indy) = 1$. We iteratively selected different components in each layer to perform the test suites and count the path value between two distinct components in different layers. It is essential to clarify that in some test suites involving two participants as agents, the same Agent instance is selected. For instance, in test suite TS_4, if ACAPY is chosen as the issuer, the holder can only be ACAPY.

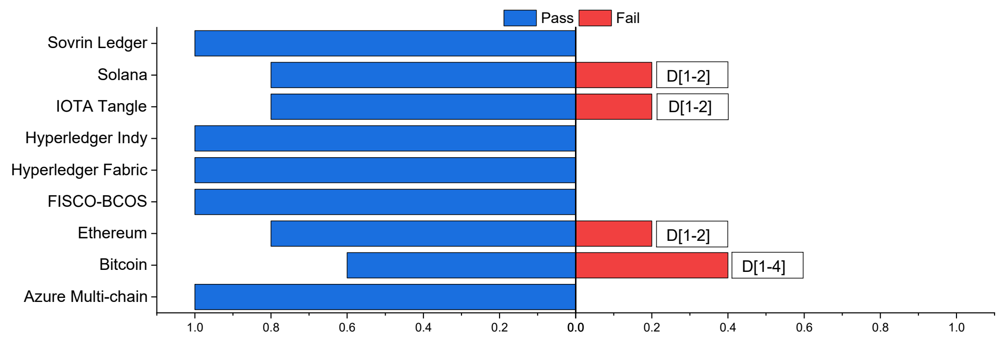


Figure 3: Test cases evaluation for DLTs. The left side represents the probability of passing test cases, while the right side depicts the probability of failed test cases corresponding to the number of failed test cases.

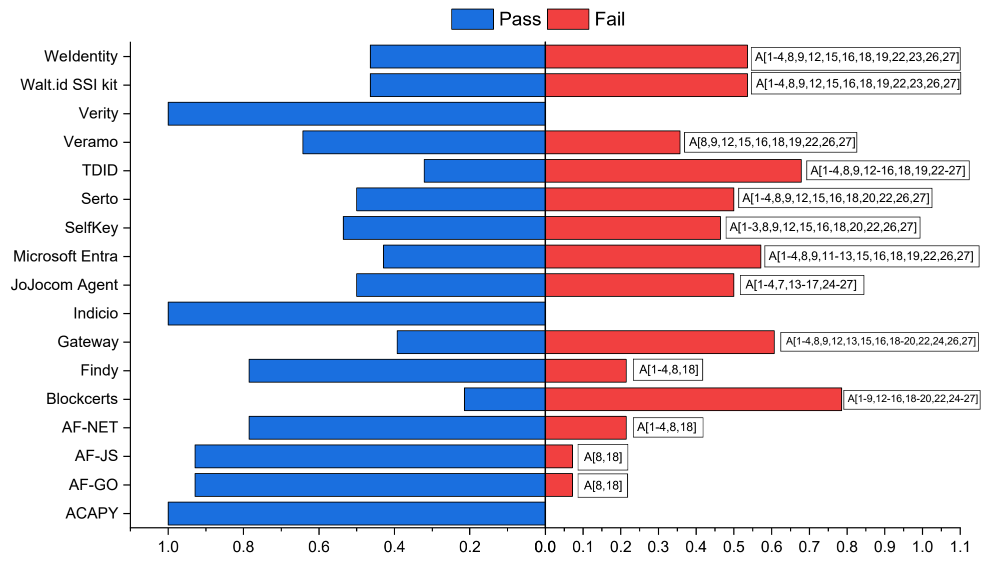


Figure 4: Test cases evaluation for agents. The left side represents the probability of passing test cases, while the right side depicts the probability of failed test cases corresponding to the number of failed test cases.

The evaluation results are shown in Figure 6. Hyperledger Indy, as a DLT implementation, specifically designed for decentralized identity management, demonstrates high compatibility with various agents and wallets due to its purpose-built architecture and feature set. As a core component of the SSI ecosystem, Indy supports a wide range of protocols and standards, enabling seamless integration and interoperability with other SSI components. Its open-source nature and modular design encourage the development of custom agents and wallets tailored to various use cases, while adhering to common SSI principles. Furthermore, Indy's active community and comprehensive documentation facilitate the adoption and implementation of the technology across different platforms. By prioritizing compatibility and extensibility, Hyperledger Indy fosters a collaborative and dynamic environment, where agents and wallets can readily interact and evolve together, advancing the broader SSI ecosystem.

5.3 Evaluation 3, Agent Interoperability

Answer to RQ5. To provide a practical demonstration of our proposed evaluation framework, we performed a group of test suites to evaluate the interoperability of different agents that serve in the same DLT as an autonomous domain.

To evaluate interoperability between agents implemented by a different instances, we utilized the test suites proposed in Table 2. The evaluation methodology is very similar to the method used in the previous section. While the only difference is that we only selected the Agent instance as both the source and destination to count the value of their path. It is essential to clarify that in some test suites involving two participants as agents, we also needed to test the situation that both the source and destination are implemented by same agents. The results are shown in Figure 7.

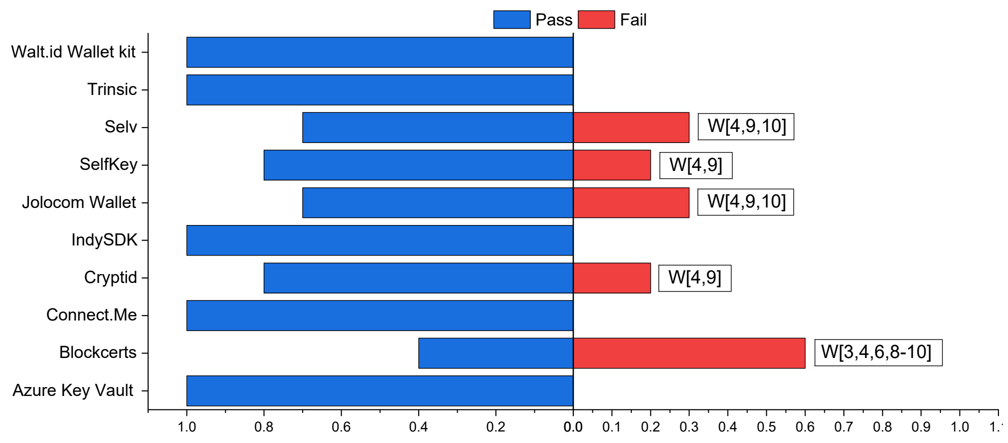


Figure 5: Test cases evaluation for wallets. The left side represents the probability of passing test cases, while the right side depicts the probability of failed test cases corresponding to the number of failed test cases.

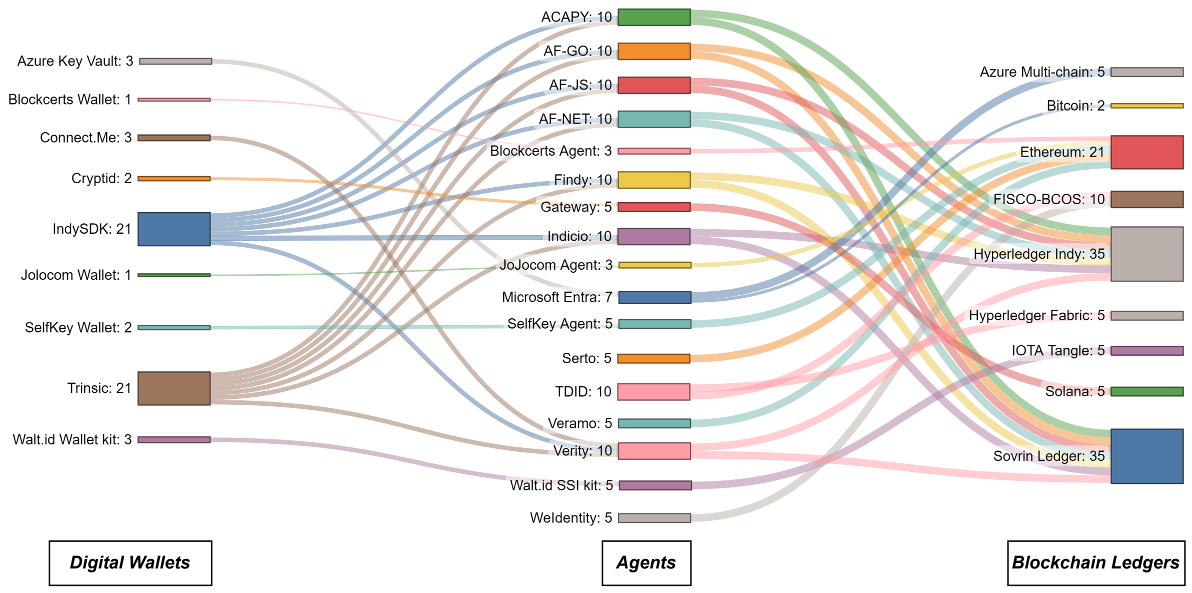


Figure 6: Test Suites Evaluation for selected SSI components. The thickness of the line represents the weight, with a thicker line indicating higher capability of the tested product.

Among the selected agents, ACAPY demonstrated the highest capability of interoperability. ACAPY is one of the wrapper instances in the Hyperledger Aries family. In addition, the other Agent instance with higher capabilities like AF-GO, AF-JS, and AF-NET, are also different wrappers that belong to the Hyperledger Aries family. Several factors contribute to the high interoperability of these products within the Aries family: (1) **Comprehensive protocol support**: Aries is built on top of Hyperledger Indy and incorporates a rich set of protocols for key management, encrypted messaging, and verifiable credentials exchange. By supporting a wide range of SSI protocols, Aries ensures seamless interaction with different SSI components. (2) **Modular architecture**: Aries features a modular architecture, which allows developers to easily integrate and extend its capabilities with other agents and wallets. This flexibility enables the creation of custom solutions tailored to specific requirements, promoting greater interoperability. (3) **Active community and collaboration**: Hyperledger Aries benefits from an active and collaborative open-source community, which continuously contributes to its development, ensuring that it remains up-to-date with the latest standards and specifications. This collaborative environment facilitates the exchange of ideas and best practices, fostering a higher degree of interoperability across the

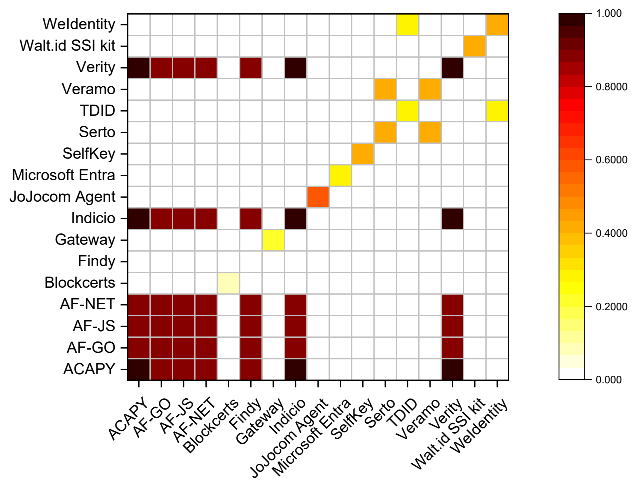


Figure 7: Test Cases for Interoperability of Agents, darker colors indicate a higher level of interoperability.

ecosystem. (4) **Wide adoption**: As one of the most widely adopted SSI frameworks, Aries has already been integrated with numerous agents and wallets, proving its compatibility in real-world scenarios. This widespread adoption speaks to its ability to work seamlessly with various SSI components. (5) **Strong documentation and support**: Comprehensive documentation and a robust support network are available for Hyperledger Aries, making it easier for developers to understand its features and implement it correctly. This level of guidance ensures that Aries can be integrated with other agents and wallets more effectively, ultimately contributing to its superior interoperability.

6 CONCLUDING REMARKS

In conclusion, our research builds upon the growing interest in adopting blockchain technology for Self-Sovereign Identity (SSI) as demonstrated by the extensive body of literature in this field. Previous studies have made significant contributions to understanding the capabilities and security aspects of various SSI frameworks, providing valuable insights into their practical applications and remaining challenges. Their work [14, 16, 33, 43, 46] provided the motivation and basis for choosing the systems that we evaluated. However, despite the numerous studies conducted in this area, there remains a gap in a comprehensive and consistent evaluation methodology that addresses the capability, compatibility, and interoperability aspects of SSI systems. Our research aimed to fill this gap by proposing a systematic methodology to assess various SSI systems, building on the foundations laid by related work.

We deliberately explored a blockchain-based SSI system's internal components and its inner workflows. We provided a comprehensive list of atomic test cases for assessing an SSI system atop a distributed ledger and cryptographic tools offered by blockchain technologies. The evaluation framework and methodologies we proposed assist researchers and developers in assessing existing blockchain-based SSI systems and evaluating their functionalities and integrabilities. The framework can be used as a baseline to compare different SSI systems and identify their strengths and weaknesses. It can also serve as a guide for developing new blockchain-based SSI systems and their components. Given the critical need to enhance information exchange and interoperability in the development of blockchain-based SSI systems, it is expected that our proposed framework can serve as a valuable reference point for the design and implementation of such systems. This will promote the integration of blockchain technologies and improve SSI's overall security and efficiency, ultimately accelerating the adoption of SSI technologies in various domains [19, 20, 37, 54, 65].

While the evaluation framework and related experiments have provided valuable insights into blockchain-based SSI systems, there are several opportunities for future research and development. We outline potential areas of exploration that will enhance our understanding of SSI systems and contribute to their ongoing improvement: (1) **Emerging technologies**: New advancements in cryptography and distributed systems offer potential avenues for enhancing the security and privacy of SSI systems. Future research will investigate the integration of emerging technologies, such as Zero-Knowledge proofs [21], secure multi-party computation [25], and homomorphic encryption [66], to further strengthen SSI systems and address any identified weaknesses or vulnerabilities. (2) **Extending the assessment of capability** into its nested layers. (3) **Enhancing compatibility**: As the adoption of distributed identity management systems become more widespread, the number of related application standards will also increase. It is crucial to ensure compatibility with the latest standards and specifications to support the growing SSI ecosystem. (4) **Standards and interoperability**: As the SSI ecosystem continues to grow, it becomes increasingly important to ensure interoperability between different SSI systems and components. Future work will focus on the development and adoption of common standards and protocols that will facilitate seamless interaction between various SSI systems and foster a more cohesive digital identity landscape. (5) **Improving applicability for automated testing**: During testing, it became apparent that many systems do not currently provide RESTful APIs [27] to interact with other systems, making it challenging to collect and count these non-RESTful APIs for a more accurate assessment. This presents an opportunity to apply more in-depth automated testing to existing test cases and suites and generate test cases automatically. (6) **Additional evaluation dimensions**: This study focused on SSI systems' core components and workflows. Future work will extend the evaluation framework by incorporating additional dimensions like scalability, performance, user experience, and economic viability. These expanded evaluation criteria will provide a more comprehensive assessment of SSI systems, enabling stakeholders to make more informed decisions about their adoption and implementation. (7) **Legal and regulatory aspects**: The implementation of SSI systems introduces new legal and regulatory compliance challenges. Future research will explore the implications of these systems in terms of data protection, privacy legislation, and identity theft prevention, as well as potential strategies for aligning SSI systems with existing and emerging legal frameworks. (8) **Addressing validity limitations**: To address current limitations to the validity of our study and obtain better feature insights, there is a need to expand the scope of the evaluation framework, incorporating additional evaluation criteria and validating the framework's applicability across a broader range of SSI systems. Engaging diverse experts and stakeholders in developing the evaluation framework will help reduce subjectivity and ensure a more comprehensive and balanced assessment. Periodically updating the framework to account for advancements in SSI technologies and trends will also enhance its long-term relevance and utility. (9) **Security and privacy evaluation**: While our current study focuses on evaluating the capability, compatibility, and interoperability of SSI systems, we acknowledge the critical importance of security and privacy protection for these systems. Such aspects require a distinct evaluation framework tailored to assess security threats and privacy enhancements comprehensively. Future research will extend our framework to include these dimensions, thereby providing a more holistic evaluation of SSI systems.

ACKNOWLEDGMENTS

We gratefully acknowledge this research is partially supported by FHWA EAR 693JJ320C000021.

REFERENCES

[1] AcaPy. Accessed in 2024. Retrived from https://github.com/hyperledger/aries-cloudagent-python.
[2] Aries Framework Go. Accessed in 2024. Retrived from https://github.com/hyperledger/aries-framework-go.
[3] Aries Framework JavaScript. Accessed in 2024. Retrived from https://github.com/hyperledger/aries-framework-javascript.
[4] Aries Framework .NET. Accessed in 2024. Retrived from https://github.com/hyperledger/aries-framework-dotnet.
[5] Oscar Avellaneda, Alan Bachmann, Abbie Barbir, Joni Brenan, Pamela Dingle, Kim Hamilton Duffy, Eve Maler, Drummond Reed, and Manu Sporny. 2019. Decentralized identity: Where did it come from and where is it going? *IEEE Communications Standards Magazine* 3, 4 (2019), 10–13.
[6] Paris Avgeriou, Philippe Kruchten, Robert L Nord, Ipek Ozkaya, and Carolyn Seaman. 2015. Reducing friction in software development. *Ieee software* 33, 1 (2015), 66–73.
[7] Tim Berners-Lee, Roy Fielding, and Larry Masinter. 2005. *Uniform resource identifier (URI): Generic syntax.* Technical Report.
[8] Bitcoin. Accessed in 2024. Retrived from https://bitcoin.org/.
[9] Blockcerts. Accessed in 2024. Retrived from https://www.blockcerts.org/.
[10] David Burnes, Marguerite DeLiema, and Lynn Langton. 2020. Risk and protective factors of identity theft victimization in the United States. *Preventive medicine reports* 17 (2020), 101058.
[11] Federal Trade Commission et al. 2021. New Data Shows FTC Received 2.2 Million Fraud Reports from Consumers in 2020.
[12] Connect.Me. Accessed in 2024. Retrived from https://connect.me/.
[13] Cryptid. Accessed in 2024. Retrived from https://cryptid.dev/.
[14] Špela Čučko, Šeila Bećirović, Aida Kamišalić, Saša Mrdović, and Muhamed Turkanović. 2022. Towards the classification of Self-Sovereign Identity properties. *Ieee Access* 10 (2022), 88306–88329.
[15] Decentralized Identity Foundation. Accessed in 2024. Retrived from https://identity.foundation/.
[16] Paul Dunphy and Fabien AP Petitcolas. 2018. A first look at identity management schemes on the blockchain. *IEEE security & privacy* 16, 4 (2018), 20–29.
[17] Samia El Haddouti and M Dafir Ech-Cherif El Kettani. 2019. Analysis of identity management systems using blockchain technology. In *International Conference on Advanced Communication Technologies and Networking (CommNet)*. IEEE, 1–7.
[18] Ethereum. Accessed in 2024. Retrived from https://ethereum.org/.
[19] Weidong Fang, Ningning Cui, Wei Chen, Wuxiong Zhang, and Yunliang Chen. 2020. A trust-based security system for data collection in smart city. *IEEE Transactions on Industrial Informatics* 17, 6 (2020), 4131–4140.
[20] Weidong Fang, Chunsheng Zhu, Mohsen Guizani, Joel JPC Rodrigues, and Wuxiong Zhang. 2023. HC-TUS: Human Cognition-based Trust Update Scheme for AI-enabled VANET. *IEEE Network* (2023).
[21] Uriel Fiege, Amos Fiat, and Adi Shamir. 1987. Zero knowledge proofs of identity. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. 210–217.
[22] Findy. Accessed in 2024. Retrived from https://findy-agent.readthedocs.io.
[23] FISCO-BCOS. Accessed in 2024. Retrived from https://fisco-bcos.org/.
[24] Gateway. Accessed in 2024. Retrived from https://www.transmute.industries/gateway/.
[25] Oded Goldreich. 1998. Secure multi-party computation. *Manuscript. Preliminary version* 78, 110 (1998).
[26] Andreas Grüner, Alexander Mühle, Tatiana Gayvoronskaya, and Christoph Meinel. 2020. A comparative analysis of trust requirements in decentralized identity management. In *Advanced Information Networking and Applications: Proceedings of the 33rd International Conference on Advanced Information Networking and Applications (AINA-2019) 33*. Springer, 200–213.
[27] Florian Haupt, Frank Leymann, Anton Scherer, and Karolina Vukojevic-Haupt. 2017. A framework for the structural analysis of REST APIs. In *2017 IEEE International Conference on Software Architecture (ICSA)*. IEEE, 55–58.
[28] Hyperledger Community. Accessed in 2024. Retrived from https://www.hyperledger.org/community.
[29] Hyperledger Fabric. Accessed in 2024. Retrived from https://www.hyperledger.org/use/fabric.
[30] Hyperledger Indy. Accessed in 2024. Retrived from https://www.hyperledger.org/use/indy.
[31] Indicio. Accessed in 2024. Retrived from https://indicio-tech.github.io/indicio-tech-stack/.
[32] JoJocom Agent. Accessed in 2024. Retrived from https://www.jojocom.io/products/jojocom-agent.
[33] Bong Gon Kim, Young-Seob Cho, Seok-Hyun Kim, Hyoungshick Kim, and Simon S Woo. 2021. A security analysis of blockchain-based did services. *IEEE Access* 9 (2021), 22894–22913.
[34] Walt.id SSI kit. Accessed in 2024. Retrived from https://walt.id/.
[35] Galia Kondova and Jörn Erbguth. 2020. Self-sovereign identity on public blockchains and the GDPR. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*. 342–345.
[36] NV Kulabukhova. 2019. Zero-knowledge proof in self-sovereign identity. In *CEUR Workshop Proceedings*, Vol. 2507. 381–385.
[37] Malni Kumarathunga, Rodrigo Neves Calheiros, and Athula Ginige. 2022. Sustainable microfinance outreach for farmers with blockchain cryptocurrency and smart contracts. *International Journal of Computer Theory and Engineering* (2022), 9–14.
[38] Sovrin Ledger. Accessed in 2024. Retrived from https://sovrin.org/.
[39] Microsoft Entra. Accessed in 2024. Retrived from https://www.microsoft.com/en-us/security/business/identity/entra.
[40] Azure Multi-chain. Accessed in 2024. Retrived from https://azure.microsoft.com/en-us/services/blockchain/multi-chain/.
[41] Nitin Naik and Paul Jenkins. 2020. Governing principles of self-sovereign identity applied to blockchain enabled privacy preserving identity management systems. In *2020 IEEE International Symposium on Systems Engineering (ISSE)*. IEEE, 1–6.
[42] Nitin Naik and Paul Jenkins. 2021. Sovrin Network for decentralized digital identity: Analysing a self-sovereign identity system based on distributed ledger technology. In *International Symposium on Systems Engineering (ISSE)*. IEEE, 1–7.
[43] Razieh Nokhbeh Zaeem, Kai Chih Chang, Teng-Chieh Huang, David Liau, Wenting Song, Aditya Tyagi, Manah Khalil, Michael Lamison, Siddharth Pandey, and K Suzanne Barber. 2021. Blockchain-Based Self-Sovereign Identity: Survey, Requirements, Use-Cases, and Comparative Study. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*. 128–135.
[44] Blaž Podgorelec, Lukas Alber, and Thomas Zefferer. 2022. What is a (digital) identity wallet? a systematic literature review. In *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 809–818.
[45] Alex Preukschat and Drummond Reed. 2021. *Self-sovereign identity.* Manning Publications.
[46] Frederico Schardong and Ricardo Custódio. 2022. Self-Sovereign Identity: A Systematic Review, Mapping and Taxonomy. *Sensors* 22, 15 (2022), 5641.
[47] Indy SDK. Accessed in 2024. Retrived from https://www.hyperledger.org/use/indy-sdk.
[48] SelfKey. Accessed in 2024. Retrived from https://selfkey.org/.
[49] Selv. Accessed in 2024. Retrived from https://selv.one/.
[50] Serto. Accessed in 2024. Retrived from https://serto.id/.
[51] Jolocom SmartWallet. Accessed in 2024. Retrived from https://jolocom.com/products/jolocom-smartwallet/.
[52] Solana. Accessed in 2024. Retrived from https://solana.com/.
[53] Ali Sunyaev and Ali Sunyaev. 2020. Distributed ledger technology. *Internet computing: Principles of distributed systems and emerging internet-based technologies* (2020), 265–299.
[54] Chatphimuk Supinyo, Pongkorn Settasompop, Plubploy Jandaeng, and Surasak Phetmanee. 2020. Ten simple rules for digital government transformation with blockchain smart contracts. *International Journal of Computer Theory and Engineering* 12, 5 (2020), 128–132.
[55] The Tangle. Accessed in 2024. Retrived from https://www.iota.org/the-tangle.
[56] TDID. Accessed in 2024. Retrived from https://www.trustdid.com/.
[57] Azure Key Vault. Accessed in 2024. Retrived from https://azure.microsoft.com/en-us/services/key-vault/.

[58] Veramo. Accessed in 2024. Retrived from https://veramo.io/.
[59] Verity. Accessed in 2024. Retrived from https://www.evernym.com/products/verity/.
[60] W3C. 2019. Verifiable Credentials Data Model 1.0. https://www.w3.org/TR/vc-data-model/
[61] W3C. 2021. Decentralized Identifiers (DIDs) v1.0. https://www.w3.org/TR/did-core/
[62] Trinsic Wallet. Accessed in 2024. Retrived from https://trinsic.id/wallet.
[63] WeIdentity. Accessed in 2024. Retrived from https://weidentity.com/.
[64] Wei Yao, Fadi P. Deek, Renita Murimi, and Guiling Wang. 2023. SoK: A Taxonomy for Critical Analysis of Consensus Mechanisms in Consortium Blockchain. *IEEE Access* (2023), 1–1. https://doi.org/10.1109/ACCESS.2023.3298675
[65] Wei Yao, Yuhong Liu, Fadi P Deek, Guiling Wang, and Ying Wu. 2023. VD-KMS: Vehicular Decentralized Key Management System for Cellular Vehicular-to-Everything Networks, A Blockchain-Based Approach. In *GLOBECOM 2023-2023 IEEE Global Communications Conference*. IEEE, 7526–7531.
[66] Xun Yi, Russell Paulet, Elisa Bertino, Xun Yi, Russell Paulet, and Elisa Bertino. 2014. *Homomorphic encryption*. Springer.