Assessing the Anonymity of DIDs in Self-Sovereing Identity

Alessio Breglia[1], Andrea De Salve[2,*], Paolo Mori[3] and Laura Ricci[1]

[1]University of Pisa, Department of Computer Science, Largo B. Pontecorvo, 3 56127 Pisa
[2]Institute of Applied Sciences and Intelligent Systems, National Research Council, Campus Universitario, 73100 Lecce
[3]Institute of Informatics and Telematics, National Research Council, Via Giuseppe Moruzzi, 1 56124 Pisa

Abstract

Self-Sovereign Identity (SSI) represents a paradigm of identity management which resolves issues of current Identity Management System (IMS) by providing to users full control over their personal data without relying on third-party providers. Decentralized Identifiers (DIDs) are essential components of SSI as they provide a new type of identifier enabling to uniquely and globally identify an entity without having to rely on a central authority. In this paper, we investigate the anonimity of DIDs used in SSI by using different sources of information to assess if these systems expose sensitive information that can be exploited to infer personal information about users. We leverage the proposed framework to evaluate the anonymity properties of a real-world DID implementation based on the Ethereum blockchain, highlighting the system's strengths and weaknesses in terms of privacy and de-anonymization risks.

Keywords

Identity Management System, Privacy, Anonymity, Blockchain, Self-Sovereign Identity

1. Introduction

In today's interconnected world, the concept of digital identity has become a fundamental part of how individuals interact online, including personal, social and professional aspects. Digital identity can be defined, therefore, as the collection of a person's attributes or characteristics that are exhibited in online interactions.

A traditional digital identity uniquely represents an individual or entity in the context of a specific digital service and it consists of two main components: *i)* the identifier that an entity uses to uniquely recognize and differentiate themselves from other entities, and *ii)* the personal information associated to the identifiers, such as personal profile information, driver's licenses, and university degrees. Typical examples of identifiers we use every day include email addresses, social network usernames, ID numbers, and product identifiers. Identity Management Systems (IMS) are responsible for managing different aspects of digital identities, including how identifiers are created, stored, accessed, and authenticated within a system. The majority of IMSs are based on a centralized or federated architecture consisting of two different types of actors: *i)* the identity providers, which act as IMS by providing identity creation, storage and authentication, and *ii)* the service providers, which request authentication decisions from identity providers in order to assert the identity of a user and retrieve their attributes. Both centralized and federated identity models suffer from significant vulnerabilities, such as single points of failure, privacy concerns related to the control of identifiers to a central authority, and dependency on third-party providers.

Self-Sovereign Identity (SSI) [1] represents a new paradigm in identity management which resolves vulnerabilities of current IMS by providing to users full control over their personal data without having to rely on a central authority. Unlike centralized or federated systems, where identities are managed by third parties, the SSI architecture enables individuals to manage and share their credentials in a secure

DLT2025: 7th Distributed Ledger Technology Workshop, June, 12-14 2025 - Pizzo, Italy
*Corresponding author.
alessiobreglia@yahoo.it (A. Breglia); andrea.desalve@isasi.cnr.it (A. De Salve); paolo.mori@iit.cnr.it (P. Mori); laura.ricci@unipi.it (L. Ricci)
0000-0003-1691-7182 (A. De Salve); 0000-0002-6618-0388 (P. Mori); 0000-0002-8179-8215 (L. Ricci)

and transparent manner. The management of identifiers in SSI is based on Decentralized Identifiers (DIDs), a new type of identifier which is cryptographically verifiable, and entirely controlled by their owners. For these reasons, DIDs promise a high level of privacy and anonymity for users, as long as identifier is only shared in a controlled way and users can create DIDs that are unlinkable across different interactions.
Despite the privacy-preserving promises of SSI, DIDs can still expose users to identification and de-anonymization risks if they are not managed properly by identity owners [2]. As for example, the metadata and the information that can be retrieved from DIDs could potentially be exploited by malicious actors to compromise their anonymity, infer sensitive information or tracking them. For this reason, it becomes crucial to investigate the ways in which anonymity of DIDs could be compromised and mitigate any potential vulnerabilities.

The goal of this paper is to investigate the anonimity of DIDs used in SSI by using different sources of information to assess whether these systems expose sensitive information that can be exploited to infer personal information about users. To achieve this, the paper proposes a framework that is responsible for the acquisition of information related to DIDs, the modeling, and the analysis of the extracted information. The framework is designed and specialized to operate on blockhain-based DIDs [3], one of the most widespread implementations of DIDs where the management and verification of DIDs is supported by blockchain. We exploited the proposed framework to assess the anonymity properties of a real DID implementation based on Ethereum blockchain, highlighting the strengths and weaknesses of the system in terms of privacy and de-anonymization risks. This paper presents several significant contributions that are outlined below:

- Analysis and identification of metadata, attributes, and any embedded information in DID documents that could inadvertently expose user identities.
- Definition of a framework for managing the acquisition of DIDs data which provides mechanisms to assess the anonymity of DIDs in Ethereum by discovering critical information leading to the deanonymization of users.
- Information about DIDs is obtained by leveraging data stored on the Ethereum blockchain as well as utilizing external data sources (e.g., Blockhain explorer, and Analytics platform).
- Modeling of DID information through graph formalism to facilitate the analysis of their relationships, patterns, and dependencies.
- The evaluation results of the proposed framework, used to assess the anonymity of a DIDs implementation in Ethereum, are described.

The outline of the paper is as follows. The foundational concepts related to SSI are described in Section 2 while the related works regarding de-anonymization and security of DIDs are presented in Section 3. The analysis and identification of the risks for anonymity in DIDs is described in 4. The architecture and functionalities of the proposed framework are described in Section 5, while Section 6 presents the results of the proposed framework on an Ethereum-based DID implementation. Finally, Section 7 draws conclusions and future improvements.

2. Background

This section introduces basic concepts related to SSI and blockchain technology.

2.1. Self-Sovereign Identity (SSI)

SSI represents an innovative paradigm in identity management, where users have full control over their personal data without having to rely on a central authority. Unlike centralized or federated systems, where identities are managed by third parties, the SSI architecture enables individuals to create, manage, and share their credentials in a secure and transparent manner. The foundation of SSI is based on 10 principles firstly defined in [4], which collectively enhance the security, control, and portability of

this new paradigm for digital identities [1]. The security of users is guaranteed by the persistence of their identities as long as users require it, by the protection of the rights of users, and by the selective disclosure of their identity information. Users have complete control over their identities, which must exist independently of the required services, and the sharing of identity information must only occur with the consent of the user. The portability of users' identities is guaranteed by the transparency of the systems and algorithms used to manage identities. This transparency ensures that users can always retrieve and access their data, making identity information usable and transportable. Furthermore, identities must not be held by any singular third-party entity.
In order to guarantee these principles, the SSI relies on three fundamental pillars: Decentralized Identifiers (DID) to represents users' identities, Verifiable Credentials (VC) and Verifiable Presentation (VP) to represents identities information, and Verifiable Data Registry to support credential/identity management and verification.

2.2. Decentralized IDentifiers (DIDs)

Decentralized Identifiers (DIDs) provide a new type of identifier which enables to uniquely and globally identifies an entity. Unlike traditional identifiers, DIDs are cryptographically verifiable and controlled by the entity (individual or organization) that owns them. Unlike traditional identifiers like e-mail, phone number, or username, which are managed and owned by centralized authorities, DIDs remain under the control of their owners as long as they possess the corresponding criptographic material. A DID, as described in W3C documentation [2], is defined as an Uniform Resource Identifier (URI) which refers to a *DID subject*, such as a person, organization, or object. Each DID is formatted according to the syntax `did:did-method:identifier`, consisting of three components: *i)* a *did* scheme, *ii)* the name of the *DID method* used to create and manage DID, and *iii)* the specific unique *identifier* provided by the corresponding DID method. Each DID is paired to a DID Document, a set of data describing the public keys and methods used to authenticate the DID subject, to interact with the DID subject by establishing a secure communication channel, to assert claims, or to create the cryptographic proofs to demonstrate control over the DID. The process of transforming a DID into its DID document is called DID resolution and it must be provided by each DID method by specifying a DID resolver, i.e., a method that fetches from a Verifiable Data Registry the necessary information for building the DID document.

As for example, the DID method *did:ethr* is a library providing DIDs under the Ethereum domain [5], where each DID corresponds to an address of an Ethereum blockchain (such as the mainnet or a testnet), resulting in DIDs having the following syntax *did:ethr:0x123456789....* By default, the *did:ethr* assumes that each Ethereum address is paired to a standard DID document. Listing 1 shows the standard DID document resulting from the resolution of the DID *did:ethr:0x123456789....* The DID document is encoded in JSON by using JSON-LD format and the *@context* attribute defines the semantics of the document. The *id* attribute specifies the DID itself. The assertion method specifies the method used by the entity to issue assertions and claims, while the authentication attribute specifies the methods used to prove control over the DID. In particular, in Listing 1 the specific properties of the assertion and authentication methods are defined by the verification method of the DID document. The verification method contains the necessary information about the type of cryptographic key, who controls the cryptographic key, and the Ethereum blockchain account from which the verification public key can be derived.

Listing 1: DID document example.

```
{
  '@context': [
    'https://www.w3.org/ns/did/v1',
    'https://w3id.org/security/suites/secp256k1recovery-2020/v2'
  ],
  id: 'did:ethr:0x123456789...',
  verificationMethod: [
    {
      id:'did:ethr:0x123456789...#controller',
```

```
      type: 'EcdsaSecp256k1RecoveryMethod2020',
      controller: 'did:ethr:0x123456789...',
      blockchainAccountId: 'eip155:1:0x123456789...'
    }
  ],
  assertionMethod: [
    'did:ethr:0x123456789...#controller'
  ],
  authentication: [
    'did:ethr:0x123456789...#controller'
  ]
}
```

2.3. Verifiable Credentials and Verifiable Presentations

A Verifiable Credential, as described in W3C documentation [6], can describe specific attributes or properties being asserted from an issuing authority about the subject of the credential. The digital signature makes VCs verifiable and tamper-evident. A VC is a set of one or more claims expressed using subject-property-value relationships. They define assertions or statements made by the issuer about the subject of the credential. Examples of verifiable credentials include digital employee identification cards, digital certificates, digital birth certificates, and digital educational certificates. VC also includes metadata that allows a verifier to understand the origin and context of the credential. Such metadata are also used to describe properties of the credential, such as the claim-issuer, the validity date and time period, a representative image, verification material, the revocation mechanism, and so on. The metadata and the credential data are signed by the claim-issuer and can be cryptographically verified. The storage of the VCs and the management of the corresponding cryptographic material is performed by specialized Wallet application.

Users can generate Verifiable Presentations from VC and then share them with a verifier to prove that they possess VCs with certain characteristics. A Verifiable Presentation (VP) is the sharing of a subset of data regarding a user, and they can express data from multiple VCs. VPs are created by the VC's owner (the user itself) and they may contain metadata that describes contextual information for proper verification (e.g., creation time, and validity) and selective disclosure information regarding the credential. Selective disclosure approaches can be used to limit the exposure of data to a verifier and to select the data of the credential to be revealed to the verifier. The VP and the corresponding metadata are cryphographically signed by the user in order to ensure its authenticity and integrity. Both VCs and VPs can be transmitted directly to the verifier, who being able to verify them without the intervention of third parties.

2.4. Blockchain data registry

Several properties of the blockchain, such as decentralization, the tamper-evident and verifiable nature of the data stored on the blockchain, align with SSI principles and make the blockchain one of the leading solutions for supporting the creation, management, and verification of DIDs. Several DID methods exploit the blockchain solution as a verifiable data registry where DIDs can be easily discovered and resolved [3]. Indeed, the blockchain accounts are based on asymmetric cryptography, and each user is able to self-create an account and use it as an identifier. The blockchain-based system ensures that the owner of the identifier is managed directly by the user and that any associated claim can be independently verified by providing cryptographic proofs regarding their ownership. The decentralized, tamper-resistant nature of blockchain ensures that information about an identifier can be modified only by the corresponding entity who controls the private key paired to the identifier. By storing cryptographic information of identities on the blockchain, the system allows these identities to be independently verified. In addition to support DID verification, the blockchain helps manage credentials, such as credential revocation, in a transparent way.

The *did:ethr* is a practical example of a DID method that uses the Ethereum blockchain as a verifiable

data registry to manage DID documents [5]. Similarly, other methods have been developed for the Bitcoin blockchain and various other distributed ledgers, each offering similar features and benefits for decentralized identity management. The *did:ethr* uses Ethereum address as specific DID identifier. For this reason, each DID identifier is paired to a pair of keys generated by using Elliptic Curve Cryptography (ECC) [7]. The private key is securely stored by the identity owner, who utilizes it as default authentication method for proving control over the DID identifier and to assert claims. The public key is utilized to generate the DID identifier and to verify the authentication of it. Any modification to the default verification methods associated with a DID identifier must be communicated to the Ethereum blockchain by sending a transaction to the smart contract *ethr-did-registry*[1]. This smart contract allows the owner of a DID identifier to specify additional attributes of a DID or verification methods. The resolution of a DID in *did:ethr* starts by querying the blockchain for any updates stored on the ethr-did-registry smart contract by the DID owner. As a result, changes made in the ethr-did-registry smart contract are directly reflected in the corresponding DID document during resolution. After retrieving the necessary information from the ethr-did-registry, the DID document is constructed off-chain.

3. Related Work

This section provides an overview of the existing scientific contributions relevant to user de-anonymization focusing on SSI systems.

De-anonymization in SSI systems presents significant risk of threats to the privacy and security of personal information. Such vulnerabilities, if breached by malicious parties, can lead to track, profile, or impersonate a user. Authors of [8] analyze the security design aspects of DID and SSI systems and claim that such a system must be designed so that data breaches are impossible, while also hindering data correlation.

The initial security and privacy considerations for DID were provided in the DID W3C specification [2]. It emphasizes the importance of applying the principles of Privacy by Design [9] to all aspects of the decentralized identifier architecture and highlights the risk of correlation attacks on DID, DID document, and DID subject.

The analisys proposed in [10] focused on the vulnerabilities that could occur when a third party who is not the owner of a DID has the authority to modify a DID document and proposes the research direction.

Naik, Grace et al. [11] proposed a systematic model to generate attack trees that can be used to perform risk analysis of a SSI system. The risk analysis evaluates possible identity theft attacks by using the attack tree. The goal of a malicious actor in this context is to obtain personal data by profiling and tracking of the user activities. In addition, the attacker can exploit the weaknesses of authentication mechanism to access information of the user wallet or theft of personal information from a background check service.

Krul, Paik et al. [12] made an analysis concerning the threats relating to SSI credential integrity and external actors. It shown that the selective disclosure of a credential supports unlinkability of DID, by allowing identity owners to control what information they reveal while maintaining anonymity.

As analyzed in [13], SSI systems that use a unique and fixed DID for each entity cannot resist to correlation analysis between different services by looking at the same public key usage. Some existing researches, such as CanDID [14], have mitigated this issue by generating distinct identifiers across different services. However, they remain vulnerable against the link analysis within a single application. Authors of [13] provides full anonymity protection by introducing a DID committee of multiple authorities which manage together with the identity owner the user's real identity by using secret sharing approach.

The SPDID system proposed in [15] exploits hardware security tecnology based on PUF to allow a user the generation of unclonable and unique information that can be used to derive cryptographic keys.

[1]Ethereum DID Registry: https://github.com/uport-project/ethr-did-registry

The authors of [16] propose to protect anonymity of a DID pseudonym by introducing a Certification Authority (CA) that exploits blind signatures to issue a privacy-preserving certificate for each pseudonym.

4. Identifying threats to SSI anonymity

Despite the privacy-preserving promises of SSI, certain elements within Decentralized Identifiers (DIDs) can expose users to re-identification and de-anonymisation risks if not properly managed. The metadata and the information of a DID document could potentially be exploited by malicious actors to compromise their anonymity, inferring also sensitive information from Verifiable Credentials (VCs) and Verifiable Presentations (VP).
For this reason, it becomes crucial to analyze these elements in SSI systems to identify and mitigate potential vulnerabilities. By understanding the ways in which anonymity could be compromised, it is possible to develop strategies to reinforce privacy protections within SSI ecosystems.

4.1. Anonimity in SSI

SSI achieves anonymity by providing the users with control and management of digital identities without relying on central authorities. This is accomplished through the use of DIDs and VCs to enable secure and private interactions. A DID is a pseudonym created and owned by the users and does not reveal any sensitive information of the entity. The DIDs are cryptographically generated by the user and stored in decentralized fashion, without being controlled by any central authority. This decentralized nature helps to protect the user's privacy and ensures that control over the identifier remains entirely with the user. In SSI systems, DIDs are used in combination with VCs, allowing users to create context-specific credentials. With the help of the *selective disclosure*, the user can select which data to disclose; thus, this ensures that only the minimum necessary data are exposed for a given transaction. Also, through the use of *zero-knowledge proofs* protocol, it allows users to verify the validity of credentials or claims without necessarily showing the actual data. The interplay of user control, selective disclosure, and cryptography techniques ensures that users can interact with other parties maintaining their privacy and anonymity, while still meeting the necessary requirements for identity verification and trust within the SSI ecosystem.

The degree of anonymity provided by DIDs can be compromised if they are not properly implemented and used. For instance, repeated use of the same DID across multiple contexts can lead to linkability, where different interactions can be correlated, compromising user anonymity. Similarly, although VCs enable selective disclosure, improper implementation or the inclusion of excessive personal information within a credential can expose users to privacy risks. In the following sections, an attacker model is developed and a deep analysis of DID, VC and VP vulnerable fields is made.

Let U denote a user in an SSI system and *PID* represent a pseudonymous identifier used by U (e.g., a DID). An adversary denoted by A aims to deanonymize U by successfully establishing a mapping M from *PID* to U, that is, $M : \mathit{PID} \rightarrow U$. This mapping is achieved by exploiting vulnerabilities in SSI mechanisms, and public information about interactions, transactions, or metadata, such that A can, with non-negligible probability, correctly associate *PID* with the real-world identity U.

4.2. Identify vulnerabilities of SSI components

In this section, the properties of DIDs are analyzed to identify potential vulnerabilities that could be exploited for de-anonymization attacks. By understanding the structure and data contained within these entities, it is possible to assess the risks associated with their use and develop strategies to mitigate them. The level of anonymity of a DIDs impacts also VC and VP, where issuers and subjects of credentials are typically represented by DIDs. In order to discover the vulnerabilities, we analyzed the fields contained within DID to understand which of them are vulnerable to de-anonymization attacks. This allowed to

Source of attack	Description	Countermeasures
ServiceEndpoint	If DID1 and DID2 have the same serviceEndpoint property, it is likely that DID1 = DID2. Also, if a serviceEndpoint shows a website link, it can be attackable	Encryption - Hashing
publicKeyPem/ publicKeyHex	If DID2 and DID2 use the same key, it implies that both identifiers are controlled by the same entity	Use DIDs associated with different keys for each interaction
AlsoKnowAs	If DID1 and DID2 have the same alsoKnowAs property, it is likely that DID1 = DID2	Encryption - Hashing
equivalentID/ canonicalID	If DID1 and DID2 have the same equivalentID property, it is likely that DID1 = DID2	Encryption - Hashing
capabilityDelegation	If DID1 and DID2 have the same capabilityDelegation property, it is likely that DID1 = DID2	Pay attention to the entities to which delegate
verificationMethod	If DID1 and DID2 have the same verificationMethod property, it is possible that DID1 = DID2	Unique methods for each DID - Encryption - Hashing
History of a DID	Having access to the history of DID document referring to the same DID can be a form of user traffic analysis (derived from the public and standardised nature of DID)	Try to provide as little useful information as possible in DID documents

Table 1
Summary of DID document vulnerabilities.

identify the fields to be considered as sources of attack, the type of attacks, and the countermeasures used to mitigate the vulnerability examined.

4.2.1. Decentralized Identifiers Vulnerabilities

This section provides a detailed description of the vulnerable fields in DID documents that could be exploited by attackers to perform de-anonymization of an entity. Table 1 summarizes the vulnerability and the countermeasure identified for each field.

serviceEndpoint The field is used in DID documents to express ways of communicating with the DID subject or associated entities. In particular, the *serviceEndpoint* property within a DID document specifies a URL or other resource location that the DID holder wants to associate with their identifier. This could be an API endpoint, a contact method, or any other type of service. If two different DIDs share the same *serviceEndpoint*, it could indicate that the same entity controls both identifiers. This weakens the anonymity of the DID holder because an attacker can compare and correlate different identities based on this shared service endpoint. To mitigate this vulnerability, it is possible to encrypt or hash with nonce the value of the *serviceEndpoint*, so that the information becomes unintelligible to unauthorized parties. When the DID owner receives an interaction request, it can decide whether or not to reveal this data to the other party.

publicKeyPem/publicKeyHex This field of a DID document contains the public key in PEM (Privacy Enhanced Mail) or Hex format. This key is used in various cryptographic operations like verifying signatures and ensuring the integrity of messages or transactions associated with the DID. Since the public key itself is not secret, the usage of the same public key on different DIDs can lead to serious security issues. An attacker can compare the public key used by different DIDs and collects the DIDs controlled by the same public key in order to link identities. To mitigate this vulnerability, the DID specification [2] suggests to use pairwise DIDs that are unique to each relationship.

alsoKnownAs This is an optional property for multiple identifiers for a DID subject, so this property is used to specify other identifiers (such as usernames, URLs, or alternative DIDs) that are associated with the same entity. If multiple DIDs have the same *alsoKnownAs* field, it suggests that they can be linked or potentially controlled by the same entity and can be exploited by an attacker to link identities.

In order to mitigate this vulnerability it is possible to encrypt or hash with a nonce the *alsoKnownAs* value, so that the information becomes unintelligible to unauthorized parties.

equivalentID/canonicalID These properties define equivalence relationships between different variants or representations of the same DID produced by the same DID method. These properties are intrinsic to the DID infrastructure and are trusted to the extent that the DID method, producer, and resolver conform to the DID specification. The field equivalentID denotes other representations of the same DID that may exist in different formats or naming conventions. The canonicalID field specifies the authoritative, preferred, or "primary" representation of the DID. However, their presence can inadvertently expose relationships between identities if the same *equivalentID* or *canonicalID* is shared across multiple DIDs. This information can be exploited by an attacker to cross-reference and de-anonymize the DID holder. To mitigate this vulnerability it is possible to encrypt or hash with nonce the value of the property, to make it not accessible for unathorized users.

capabilityDelegation This property is used to specify a mechanism that might be used by the DID subject to delegate a cryptographic capability to another party, such as delegating the authority to access a specific HTTP API to a subordinate. An example of when this property is useful is when a DID holder chooses to delegate their capability to access a protected HTTP API to a party other than themselves. If multiple DIDs delegate capabilities to the same entity or if the *capabilityDelegation* field is identical across different DIDs, it may suggest that these identities are related or controlled by a common entity. Ensuring access to *capabilityDelegation* only to trusted entities reduces the risk of de-anonymisation attacks.

verificationMethod A DID document can express verification methods, such as cryptographic public keys, which can be used to authenticate or authorize interactions with the DID subject or associated parties. Inside the *verificationMethod* field, *Verification Material* is present: it is any information that is used by a process that applies a verification method. The type of verification method is expected to be used to determine its compatibility with such processes. For example, a cryptographic public verification method with respect to a digital signature verifies that the signer could use the associated cryptographic private key. In addition, sharing the same *verificationMethod* across different DIDs can lead to identity correlation, depending on how many DIDs use the same *verificationMethod*. Furthermore, if the same cryptographic key or method is used by multiple DIDs, an attacker can deduce that these identities are linked, thereby compromising the anonymity of the user. Implementing the required *verificationMethod* properties or using a *verificationMethod* which is widely used helps prevent correlation attacks. By doing so, the linkability of identities is minimized, preserving the anonymity and privacy of the DID holder.

History of a DID When a blockchain-based DID used, its history is typically recorded as a series of transactions or updates on the blockchain. Access to the complete history of a DID can be a significant privacy risk: an attacker performing traffic analysis might deduce patterns or behaviors based on transactions. Limiting the use of a DID to a specific activity can effectively reduce the risk of user traffic analysis and de-anonymisation attacks by using public blockchain data.

5. A framework for analyzing the anonymity of Ethereum-based DIDs

De-anonymization attacks in Ethereum DID ecosystem are possible through a systematic approach starting from analyzing DID documents, transactions and interactions, to create a DID profile. Figure 1 shows the workflow used to analyze the anonymity of DID by the proposed framework. The main phases of the framework are the data acquisition, in which all the necessary data are taken, the DID analysis, in which the DID documents and their fields are analysed based on the vunerabilities identified

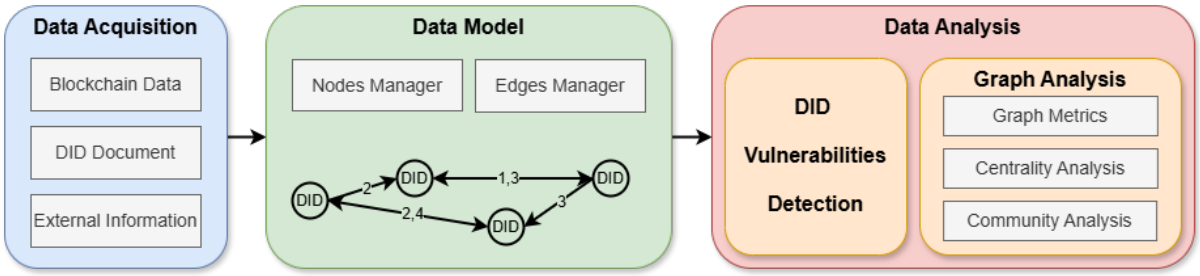


Figure 1: Architecture of the proposed framework.

in Section 4, and the graph analysis, in which the data enriched with graph formalism to perform different tasks, such as the community analysis.

5.1. Data acquisition

Firstly, the address of smart contracts responsible for managing DIDs within the Ethereum blockchain are identified. As for example, the ethr-did-registry smart contract provided by the *did:ethr* DID method operates at the address *0xdCa7EF03e98e0DC2B855bE647C39ABe984fcF21B*. The access to the Ethereum blockchain is performed in order to read all transactions which invoked the ethr-did-registry smart contract. These transactions are directly linked to DIDs because Etherum addresses are used as DID identifiers and only the DID owner can perform transactions on the ethr-did-registry smart contract. For this reason, we retrieved values of the sender address specified by the from field of each transaction. Based on the specification of the *did:ethr* method, the Ethereum address of the sender serves as unique DID identifier, allowing the framework resolve each DID into the corresponding DID document.
Another source of information that is important to understand the behavior and interactions of blockchain-based DIDs are transactions initiated by DIDs using the same blockchain address specified in the DID document. For this reason, for every DID identified in the ethr-did-registry, all the transactions in which DIDs are involved are retrieved from the Ethereum blockchain.

In addition to internal blockchain data, the data acquisition component extracts information related to a DID identifier from external sources (such as block explorer and analytics platform [17]). In particular, we collected from Etherscan[2] the DID funder and the DID nametag. The DID nametag is an alternative human-readable identifier provided by Etherscan which can be used to specify the name or alias of the entity. The DID funder is an information obtained by Etherscan by examining the first transaction perfomed by a DID and it indicates whether a DID is funded by major cryptocurrency platforms such as Coinbase, Kraken, or Decentralized Exchange (DEX). This information can be used in the data analysis to establish connections between DIDs and well-known financial entities, potentially revealing affiliations or funding sources.

5.2. Graph-based modeling

The collected data are transformed by the framework into a graph model to enable de-anonymization analysis. Each unique Ethereum address collected during data acquisition is represented as a node in the graph, and an edge is created between two nodes whenever a transaction occurs between the corresponding addresses. The edges are directed, reflecting the direction of the transaction from the source DID to the target DID, and the weight of each edge is determined by the number of times a particular operation occurred between the source and target DIDs. If multiple transactions of the same operation occur between the same pair of addresses, the counts are aggregated to reflect the total frequency, indicating how often interactions happened between two specific users.
Each node in the graph is enriched with a set of attributes derived from both blockchain data and

[2]Etherscan: https://etherscan.io/

external services. These attributes include the DID document, DID nametag, DID funder, and other relevant information related to a DID.

5.3. Data analysis

The data analysis component conducts vulnerability detection and graph analysis on the model to identify potential risks and patterns of de-anonymization. It examines how these DIDs interact within different communities and the types of operations performed by each.

DID Vulnerabilities Detection This component indicates vulnerable fields from a DID document based on the analysis presented in Section 4, such as serviceEndpoint, alsoKnownAs or other metadata informations that might link a DID to a real-world identity or a recognizable entity. In addition to identify vulnerabilities in the DID document, the component is able to evaluate the distribution of the operations made by DIDs on the DID document by analyzing the transactions on the ethr-did-registry smart contract.

Graph Analysis Graph analysis is chosen because it serves as a powerful tool for understanding the unique attributes and behaviors associated with each DID. By constructing and analyzing a graph representing the interactions and relationships of DIDs, it becomes possible to uncover specific patterns and characteristics that are distinctive to each DID. These insights enable the creation of detailed profiles, facilitating the process of de-anonymization by associating behaviors, operations, and connections to potential real-world entities or contexts.
For this reason, the framework is able to compute general graph metrics: such as measures the number of nodes and edges, how closely the nodes in a graph are connected (density), assesses the degree to which nodes in a graph tend to cluster together (Clustering coefficient), quantifies the strength of division of a graph into communities (Modularity), measures the uniformity of a graph's connections within communities (Homogeneity), examines the tendency of nodes to connect with similar nodes (Assortativity).

Another important technique provided by the framework for the analysis of the DIDs is based on centrality. Centrality measures allow to identify key entities or operations, and their role within the system. For this reason, the framework includes various measures, such as degree, betweenness, and closeness centrality that indicate the importance or influence of a node within the graph.

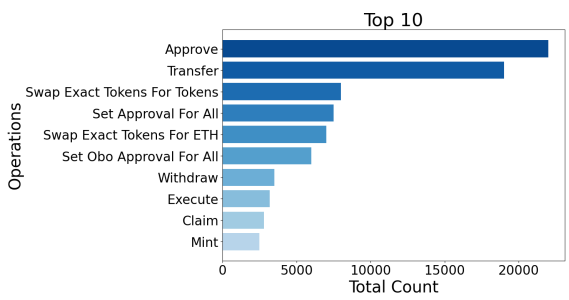
The proposed framework also integrates a community detection algorithm based Louvain method which is able to group the DIDs into distinct communities. The community analysis of DIDs can provide important information about the intractions and behavior of DIDs and can be used to extract informations the entities with which a DID has frequently interacted and belongs to the same community. In addition, community detection allows to identify the type of activities a DID is involved in and the platforms or entities it interacts with. For example, if a DID is involved in particular tokens, it is possible to infer some behavioural information. Similarly, investments in niche tokens or markets could reveal personal interests or financial positions.

By combining the blockchain data (transactions, token), external data (such as nametage and funders), and potential vulnerabilities in the DID document, it is possible to build a detailed profile of a DID. Such profile can be enriched by using information related to the behavior, associations, and metadata.

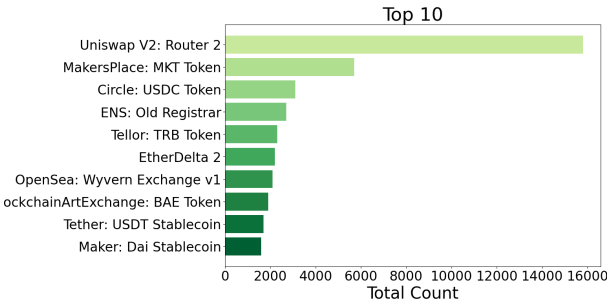
6. Experimental Results

This section presents the results of the analysis conducted on Ethereum DIDs ecosystem by exploiting an implementation of the proposed framework[3].

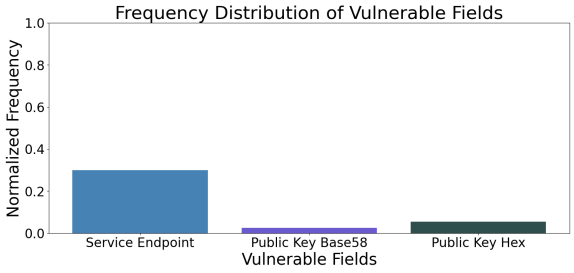
[3]Available at https://github.com/Breian/DLT2025---Assessing-the-Anonymity-of-DIDs-in-Self-Sovereing-Identity.git.



(a) DID's operations



(b) Targets of DID's operations



(c) DIDs verification methods

Figure 2: Distribution of the DIDs verification methods, main DID's operations, and main targets of DID's operations

6.1. Results of DID Vulnerabilities analysis

We present the results of the DID Vulnerabilities analysis by analyzing the distribution of the vulnerable fields in DID document and operation obtained by a total of 31714 transactions performed on the ethr-did-registry smart contract by DID owners.

6.1.1. DID documents

The transactions list allow the resolution of 3613 distinct DID documents. The results reveal that attributes *versionId* and *updated* of a DID document occur very frequently (3259 DID documents specify these attributes), suggesting that most of the DID owners have versioned their DID document and have updated them at least once since their creation. The DID vulnerabilities analysis also reveals the presence of the attributes *serviceEndpoint*, *publicKeyBase58* and *publicKeyHex*, that could potentially expose sensitive information of 10 different DID documents over 3613. The distribution of these fields within the collected dataset is shown in Figure 2c.

The public key fields *publicKeyBase58* or *publicKeyHex* are used by few DID documents in order to specify a different authentication key based on either *EdDSA* algorithm [18] or compressed *ECDSA* *secp256* [19]. In addition, the specified keys are unique and it is not possible to correlate the DID with other DID documents using the same keys. The *serviceEndpoint* attribute describes the service endpoint and it reveals different types of information about the DID owner. All the DID documents have specified as a value for *serviceEndpoint* an unencrypted value, while some DID documents (2) specify an authenticated value (using JWT [20]). The majority of DID documents (6) use a publicly accessible web url or an email as *serviceEndpoint*. However, the website url and email are vulnerable because an adversary can use it to perform a phishing attack or to access relevant information for identifying the DID owner. Indeed, such website urls reveal relationships between the DID owner and other entities, such as projects (uPort), a web resource on IPFS [21], an application (HerokuApp), a cloud provider (https://talao.co), or a web service (https://travel-rule-protocols.notabene.dev/). Some DID documents (2) specify as *serviceEndpoint* a JWT that can protect the authenticity and integrity of data. However, data confidentiality is not guaranteed because the DID vulnerabilities analysis reveal that such JWT represents a verifiable presentation of a Texas-NotaryCredential, issued by a recognized entity named

Table 2
Graph general metrics

Metrics	Value
Density	0.000148
Clustering Coefficient	2.00e-05
Assortativity Coefficient	-0.276
Modularity	0.586
Homogeneity	0.657

BillyCaseworker.

The vulnerabilities of the verification method across the DID documents are identified by examining the *verificationMethod* field of each DID and checking if one method is similar or identical to other DID documents. The verification methods used in DID documents are *EcdsaSecp256k1RecoveryMethod2020*, *X25519KeyAgreementKey2019*, and *EcdsaSecp256k1VerificationKey2019*. The *EcdsaSecp256k1RecoveryMethod2020* verification method is used by 99% of the analysed DIDs and it is an ECDSA algorithm based on the Secp256k1 curve. The method *X25519KeyAgreementKey2019* is used by only one DID to specify a elliptic curve key agreement algorithm while *EcdsaSecp256k1VerificationKey2019* is used by 2 DIDs to specify another ECDSA method using based on *secp256k1* curve. The analysis of the verificationMethod reveals that the majority of DID documents do not introduce any vulnerabilities and do not share other relevant information.

6.1.2. DID operations

Another information provided by the DID vulnerability analysis is the type of transactions executed by DID owner on the Ethereum Blockchain and the entity to which this transaction is intended. Figure 2a shows the top 10 most executed operations by DID owners. The *approve* operation is the most frequently performed across all DIDs, allowing the DID to manage tokens by granting permission for another entity (contract or individual) to transfer a specified amount of tokens on their behalf. The *transfer* operation represents asset movement and refers to the direct transfer of tokens or assets from a DID. The *Swap Exact Tokens For Tokens* and *Swap Exact Tokens For ETH* are used to manage liquidity by exchanging one type of token for another or for Ether. The *Set Approval For All* is used by a DID to authorize an operator to manage all of their assets while the *Set Obo Approval For All* operation is used to set approval for all actions on behalf of another party. Operations like *Withdraw*, *Execute* and *Claim* are less frequent for DIDs and they are used for the execution of transactions or functions in smart contracts, for claiming rewards, tokens, or assets that may have been locked or earned through a DeFi protocol. The *mint* operation involves creating new tokens or assets and it suggests that some DIDs have responsibility in token creation.

Figure 2b shows the 10 most frequent targets of the operations made by DIDs. In general, trading platforms and Exchangers (such as Uniswap) are the targets of the most part of the transaction performed by DIDs. The most frequent target of transactions is *Router 2* smart contract, which is used to facilitate liquidity management within the Uniswap V2 protocol. Another traiding platform that is frequently used by DIDs is the EtherDelta platform. The *MKT Token* smart contract indicates a considerable amount of activity related to token or operations associated with the MakersPlace platform. Similarly, the *BAE Token* and the *Wyvern Exchange* are exchangers related to the blockchain crypto-art ecosystem and they refer to BlockchainArtExchange and OpenSea (two of the most important NFT marketplace platforms). Stablecoins (sush as USDT, USDC, and DAI) represent another main target for several DID owners, which use them as a liquidity, trading, and exchange medium. The Ethereum Name Service (ENS) is a distributed naming system which allows to map human-readable names to machine-readable identifiers to Ethereum entities (e.g., addresses or cryptocurrency). Finally, another important target for DID activities is the *TRB Token* smart contract provided by Tellor, which is focused on providing oracle-related token. The presence of OpenSea and TRB Token suggests that NFT marketplaces and

Table 3
Summary of the characteristics of DIDs communities where VS= Very small, S=Small, M=Medium, L=Low, LA=Large, VL=Very Low, HN=High Negative, N=Negative, Z=Zero

Communities	#Nodes - #Edges				Density		Assortativity		Clust. Coeff.		
	VS	S	M	LA	VL	L	HN	N	Z	VL	L
$c_{12}, c_{15}, c_{17}, c_{19}, c_{23}, c_{24}, c_{25}, c_{27}, c_{28}, c_{30}$	•					•	•		•		
c_1, c_4, c_{11}, c_{13}				•	•			•		•	
c_{21}, c_{26}, c_{29}		•				•	•		•		
c_2, c_9, c_{14}			•		•			•		•	
c_6, c_7			•		•		•		•		
c_5, c_{16}			•		•		•			•	
c_8			•		•			•	•		
c_{18}		•				•		•			•
c_{22}				•	•		•				•
c_{20}	•					•		•	•		
c_{10}	•					•		•		•	

oracle services are an important part of the observed transactions.

6.2. Graph-based Analysis

The graph-based analysis considers the graph of transactions performed by each DID towards other DIDs or Ethereum addresses and consists of 31714 nodes and 74654 edges. Table 2 shows the number of nodes, the number of edges, the density, the assortativity, and the clustering coefficient (Clust. Coeff.) of each community. The metrics suggest that the graph is disassortative and sparse with moderate strong community structures, where most interactions occur within distinct groups, and a few highly connected nodes serve as hubs interacting with many smaller, less connected entities. In particular, the low density value means that the network is sparsely connected,and the low value of the clustering coefficient indicates that most DIDs interact individually with other entities without forming interconnected groups, while homogeneity indicates that most connections are internal to communities.

The most central DIDs in terms of *degree*, *betweenness*, and *closeness* centrality are also evaluated in the graph, showing the different roles in the Ethereum blockchain ecosystem. The most central DIDs belong to key participants or infrastructure entities enabling important functions, such as liquidity management, governance operations, token transfer, and asset control. Several of the top DIDs, particularly those with high closeness and betweenness centrality, are essential to the DeFi ecosystem, like *Uniswap V2: Router 2*, *Circle: USDC Token*, *Maker: Dai Stablecoin* and *CH DAO: Deployer*, which helps to manage governance tokens, liquidity, and the decision-making processes of DAOs. Instead, DIDs with high degree centrality demonstrate high engagement in NFT minting and token management, like *MakersPlace: MKT Token*, which is a significant actor in the NFT marketplace, and *coopahtroopa.eth*, that is another active participant in the DeFi and NFT space.

6.2.1. Communities Analysis

The results of the community detection performed with the Louvain algorithm identified a total number of 30 communities ($c_1, \ldots, c_{30}$) in the graph. For each community, we computed the density, assortativity, and clustering coefficient to better understand the structural characteristics of each community. Table 3 summarizes the structure of the communities based on the evaluated metrics.
The majority of communities ($c_{12}, c_{15}, c_{17}, c_{19}, c_{20}, c_{23}, c_{24}, c_{25}, c_{27}, c_{28}, c_{30}$) consist of few participants and interactions (at most 100 nodes and 100 edges), have sparse connections and hierarchical structure resulting from both a very low densitity value between 0 and 0.0073, a high nevative assortativity value between -1 and -0.7, and a zero cluestering coefficient. Some of these very small communities are centered around specific assets, such as the community c_{19} which is centered around the Falcon Project: Voucher and community c_{25}, which is centered on the minting of NFTs and other digital assets.

Communities c_{21}, c_{26}, c_{29}, and c_{18} have a small number of participants and interactions (i.e., the number of nodes and edges of each community is included between 100 and 500) and expose a low value of density. In particular, the high negative value of assortativity and the zero clustering coefficient suggest that the communities c_{21}, c_{26}, and c_{29} are hierarchically structured. In contrast, community c_{18} exposes a negative value (between -0.7 and -0.3) of assortativity and a low cluestering coefficient (between 0.001 and 0.01).
Another large group of communities with significant participants and interactions are $c_2, c_5, c_6, c_7, c_8, c_9, c_{14}$, and c_{16}. The number of nodes in such communities ranges from 500 to 2000 while the number of edges ranges from 1000 to 5000. All these communities expose a very low value of density and a hierarchical structure, while only half of them (c_2, c_8, c_9, c_{14}) expose a very low value of clustering (below 0.001) suggesting sparse local interconnectedness and rare formation of triangles. Most of these communities demonstrate centralized patterns, in which one or a few dominant DIDs are the central hub of most operations. In general, these DIDs execute central operations, such as transfer, approval, and governance actions, often against DeFi platforms, NFT markets, or other DApps. The communities c_1, c_4, c_{11}, c_{13}, and c_{22} consist of a large number of participants and interactions (the number of nodes is greater than 2000 and the number of nodes is greater than 5000). The low density value and the positive value of the clustering coefficient indicate the presence of some cohesive subgroups in each community. In particular, the cluestering coefficient of the community c_{22} ranges between 0.001 and 0.1, where a central DID exposes the role in governance management by performing the majority of transfers and confirm/exec transactions.

The analysis of structural metrics (such as centrality) combined with the underlying community structure, provides valuable insights into the behavioral patterns and functional roles of entities within the network. These metrics help uncover how entities interact, whether they act as central coordinators, peripheral participants, or bridges between subgroups, and it is particularly useful in contexts of de-anonymization.

7. Conclusions and future work

In this paper, a framework for analyzing the anonymity of DIDs has been proposed. The approach focused on blockchain-based DIDs and leverages transaction data, DID document analysis, and graph-based analysis to reveal hidden correlations between user activities and personal attributes. The proposed framework was used to assess the anonymity properties of a DID implementation based on Ethereum, allowing to successfully derive information related to a real identities of 30 DIDs.

In several cases, DIDs are immediately de-anonymised through explicit information obtained from the blockchain or from external sources of information (e.g., block explorer platform). Collectively, these results illustrate that while SSI in Ethereum is designed to provide users with complete control over their identities, the transparency of blockchain transaction data, when combined with graph analytics and community detection methods, can reveal information of the entity behind a DID.
As a direction for future research, it is essential to refine the proposed methodology by enhancing its analytical capabilities through the integration of AI approach for inferring information from DIDs and explore the application of a framework to cross-chain schenario where DIDs belong to different blockchains.

Acknowledgments

This work was partially supported by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] A. Tobin, D. Reed, The inevitable rise of self-sovereign identity, 2016. URL: https://sovrin.org/wp-content/uploads/2018/03/The-Inevitable-Rise-of-Self-Sovereign-Identity.pdf.

[2] M. Sporny, D. Longley, M. Sabadello, D. Reed, O. Steele, C. Allen, Decentralized identifiers (dids) v1.0 - core architecture, data model, and representations, 2022. URL: https://www.w3.org/TR/did-1.0/.

[3] Q. Stokkink, J. Pouwelse, Deployment of a blockchain-based self-sovereign identity, in: 2018 IEEE international conference on Internet of Things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData), IEEE, 2018, pp. 1336–1342.

[4] C. Allen, The path to self-sovereign identity, 2016. URL: https://www.lifewithalacrity.com/article/the-path-to-self-soverereign-identity/.

[5] Uport, Ethr-did library, 2025. URL: https://github.com/uport-project/ethr-did.

[6] M. Sporny, D. Longley, D. Chadwick, Verifiable credentials data model 1.0 - expressing verifiable information on the web, 2019. URL: https://www.w3.org/TR/vc-data-model-1.0/.

[7] G. Wood, et al., Ethereum: A secure decentralised generalised transaction ledger, Ethereum project yellow paper 151 (2014) 1–32.

[8] C. N. Butincu, A. Alexandrescu, Design aspects of decentralized identifiers and self-sovereign identity systems, IEEE Access 12 (2024) 60928–60942.

[9] A. Cavoukian, et al., Privacy by design: The 7 foundational principles, Information and privacy commissioner of Ontario, Canada 5 (2009) 12.

[10] M.-H. Rhie, K.-H. Kim, D. Hwang, K.-H. Kim, Vulnerability Analysis of DID Document's Updating Process in the Decentralized Identifier Systems , in: 2021 International Conference on Information Networking (ICOIN), IEEE Computer Society, Los Alamitos, CA, USA, 2021, pp. 517–520.

[11] N. Naik, P. Grace, P. Jenkins, An attack tree based risk analysis method for investigating attacks and facilitating their mitigations in self-sovereign identity, in: 2021 IEEE Symposium Series on Computational Intelligence (SSCI), 2021, pp. 1–8.

[12] E. Krul, H.-y. Paik, S. Ruj, S. S. Kanhere, Sok: Trusting self-sovereign identity, Proceedings on Privacy Enhancing Technologies 2024 (2024) 297–313.

[13] Y. Liu, Z. Zhao, F. Ran, X. Lin, D. Li, Z. Guan, et al., Fully anonymous decentralized identity supporting threshold traceability with practical blockchain, in: THE WEB CONFERENCE 2025, 2005.

[14] D. Maram, H. Malvai, F. Zhang, N. Jean-Louis, A. Frolov, T. Kell, T. Lobban, C. Moy, A. Juels, A. Miller, Candid: Can-do decentralized identity with legacy compatibility, sybil-resistance, and accountability, in: 2021 IEEE Symposium on Security and Privacy (SP), IEEE, 2021, pp. 1348–1366.

[15] Y. He, W. Fan, K. Inoue, Spdid: A secure and privacy-preserving decentralized identity utilizing blockchain and puf, in: 2024 IEEE 23rd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), IEEE, 2024, pp. 1622–1631.

[16] J. S. Shin, S. Lee, S. Choi, M. Jo, S.-H. Lee, A new distributed, decentralized privacy-preserving id registration system, IEEE Communications Magazine 59 (2021) 138–144.

[17] A. De Salve, A. Brighente, M. Conti, Edit: A data inspection tool for smart contracts temporal behavior modeling and prediction, Future Generation Computer Systems 154 (2024) 413–425.

[18] S. Josefsson, I. Liusvaara, Edwards-curve digital signature algorithm (EdDSA), Technical Report, 2017.

[19] D. Johnson, A. Menezes, S. Vanstone, The elliptic curve digital signature algorithm (ecdsa), International journal of information security 1 (2001) 36–63.

[20] M. Jones, J. Bradley, N. Sakimura, Json web token (jwt), Technical Report, 2015.

[21] J. Benet, Ipfs-content addressed, versioned, p2p file system, arXiv preprint arXiv:1407.3561 (2014).