Methods for Decentralized Identities: Evaluation and Insights

Walid Fdhila[1,2], Nicholas Stifter[1,2], Kristian Kostal[3], Cihan Saglam[4], and Markus Sabadello[4]

[1] Secure Business Austria (SBA-Research), Austria.
[2] University of Vienna, Austria
[3] FIIT, Slovak University of Technology in Bratislava, Slovakia
[4] DanubeTech GmbH, Austria.

Abstract. Recent technological shifts have pressured businesses to reshape the way they operate and transact. At the hart of this restructuring, identity management established itself as an essential building block in both B2C and B2B business models. Trustworthy identities may refer to customers, businesses, suppliers or assets, and enable trusted communications between different actors. Unfortunately, traditional identity management systems rely on centralized architectures and trust in third party services. With the inception of blockchain technology, new methods for managing identity emerged, which promise better decentralization and self-sovereignty. This paper provides an evaluation of a selection of distributed identity methods, and analyzes their properties based on the categorization specified in the W3C recommendation rubric.

Keywords: Blockchain · Distributed Identity · Self-sovereign Identity · DID Method.

1 Introduction

In today's internet, organizations such as Google, Facebook or Amazon centrally manage and control vast amounts of cross-correlating data about individuals and their identities. An already diminished trust in such centralized systems by its users is further brought into question, as recent breaches have exposed their private data on a massive scale, urging the need for new *decentralized* methods that give individuals full control back over their data.
By providing the necessary infrastructure and renewed interest in Byzantine Fault Tolerance (BFT) [11], the advent of blockchain technology paved the way for such new decentralized methods for establishing trustworthy distributed identities that do not rely on a central entity serving as a single point of trust. This approach is called self-Sovereign Identity (SSI), in which entities or individuals become their own identity providers, thus creating and controlling one or multiple (i) decentralized identifiers (DIDs), and (ii) verifiable credentials (VCs) [6]. (i) A DID is a unique identifier, usually associated with a DID document (also called continuation document) that specifies cryptographic material, verification

methods and services essential for proving ownership of the DID and trustworthy communication with the DID owner. (ii) Verifiable credentials are identity attributes and assertions about a specific subject issued by an identity provider. In contrast to traditional credentials, a relying party (third party service) can check the validity of a VC without having to interact with the issuer. A DID method, on the other hand, defines how a DID can be created, resolved, updated and revoked. Currently, there exist over one hundred DID methods that rely on different architectural designs or infrastructures. Therefore, developing a use case that employs decentralized identifiers requires a good understanding of the properties of such methods and what they offer in terms of governance, security or operation. In this paper, we provide a qualitative evaluation of six DID methods following the guidelines of the W3C DID method rubric.[5]
The remainder of the paper is structured as follows. Section 2 introduces basic concepts, while Section 3 outlines the methodology. Section 4 evaluates the different DID methods, Section 5 discusses the results and concludes the paper.

2 Background

Decentralized Identifiers (DIDs). DIDs are unique identifiers whose purpose is to ensure trustworthy and persistent communication channels between entities. In contrast to common identifiers such as URIs, phone numbers or social media identifiers, which are issued and controlled by third parties, DIDs enable individuals and organizations to issue and control their own identifiers. DIDs, as specified in the W3C recommendation draft[6], are strings that have the following format: $did :< did\_method >:< method\_specific\_identifier >$, where *did* is the prefix, *did_method* refers to the method specification that defines the precise operations for creating, resolving, updating and revoking specific DIDs, and $method\_specific\_identifier$ is a unique identifier within the method. An example of a DID created using the *btcr* method[7] on the Bitcoin blockchain would be: $did : btcr : 8kyt - fzzq - qpqq - ljsc - 5l$

DID Document. Similar to DNS resolution where a URL is provided as an input and the corresponding IP address is returned as output, DID resolution takes the DID as input and returns a DID document as output. The latter contains, among others, cryptographic material (public keys for authentication, authorization, and interaction), verification methods for proving ownership of the DID, and service endpoints for enabling trusted communication with the subject, for instance to exchange verifiable credentials.

DID Method. The diversity in the blockchain ecosystem led to a plethora of different methods for creating and resolving DIDs. Although most of these methods comply with the W3C DID specification, each of them comes with specific properties and offers different guarantees depending on the underlying technology or

[5] This work expands upon a technical report that evaluates DID method specifications [4] which was conducted by the authors.
[6] See: https://www.w3.org/TR/did-core
[7] See: https://w3c-ccg.github.io/didm-btcr/

governance framework. As such, each DID method specifies how corresponding DIDs are **c**reated, **r**esolved, **u**pdated and **d**eactivated (CRUD).

3 Methodology

3.1 DID Method Selection

At the time of writing, there exist over 100 DID methods that differ in various aspects such as (i) the underlying infrastructure (ii) governance, (iii) operation, and (iv) security. Unfortunately, a large number of these methods are either conceptual designs, unimplemented proposals or stale projects that are no longer maintained. Additionally, the focus is often placed on operational elements such as CRUD, while the remaining aspects of the method receive considerably less attention and documentation, rendering an evaluation difficult.

Based on the authors' expertise in the domain of SSI and their involvement in the implementation of the universal registrar and resolver for DIDs[8,9], a set of 6 DID methods, that cover different architectural designs, were selected and evaluated: (i) blockchain-based (ii) non blockchain-based, (iii) public permissioned, (iv) public permissionless and (v) pairwise DIDs. The selection, namely {DID:BTCR, DID:V1, DID:ETHR, DID:SOV, DID:WEB, DID:PEER} offers sufficient documentation and implementation details for a fair evaluation and covers approaches that are currently well received by the SSI community.

3.2 Evaluation Process

In this paper a qualitative evaluation is performed using the guidelines specified in the W3C DID method Rubric V1.0 (06 Jan. 2021)[10] and additional criteria derived from the principles of SSI [8]. The paper, therefore, provides both a comprehensive and comparative study of the DID methods and for each evaluation criteria considers three overlapping dimensions as follows:

(i) *Network* is the underlying communication layer, i.e., how and with whom users need to communicate to invoke the operations of the method.
(ii) *Registry* is a given instance of recorded state changes, managed according to the specification, using the communication layer.
(iii) *Specification* is the governing document of the method that defines and outlines how a particular method implements the required and any optional components of the DID core specification.

Evaluation criteria. The criteria are grouped into four categories, each focusing on a specific aspect of the method:

[8] See: https://dev.uniresolver.io/
[9] See: https://github.com/decentralized-identity/universal-registrar/
[10] See: https://w3c.github.io/did-rubric/

(1) *Rulemaking* captures the degree of decentralization in the governance of the DID method. It covers who can define the rules and how they are defined with respect to each of the aforementioned dimensions. For instance, the economic interest behind a DID method can impact its centralization if the goal is to support the interest of a certain group.
(2) *Operation* focuses on the CRUD operations and evaluates how the rules are executed. It also addresses the *openness* of the operation, i.e., whether it is restricted to a select group (permissioned) or open to participation by anyone (permissionless). Permissioned operation can impact the availability of the network to various participants, which affects inclusivity with regard to underserved or vulnerable populations. It may also expose the permission giver to legal ramifications.
(3) *Security* covers potential attack vectors against both, the integrity and correctness, as well as the privacy and self-sovereignty of users, that can arise through the method design choices and employed technologies.
(4) *Implementation* touches on aspects and challenges regarding an actual implementation and utilization of the corresponding DID method in practice.

The evaluation is conducted by five experts with diverse technical and theoretic backgrounds in distributed ledger technology in general and self-sovereign identity in particular. Some of the evaluators are also involved in standardization efforts by W3C for decentralized identifiers and verifiable credentials, in addition to projects for the implementation of a universal DiD registrar and resolver that supports around 50 DiD methods. As aforementioned, the selected DiD methods were chosen to cover different architectural designs and rely on various infrastructures that obey to different governance rules.

4 Evaluated DID Methods

4.1 did:btcr

Description DID:btcr uses transactions on the Bitcoin blockchain for registering, updating and revoking identities. The DID corresponds to the transaction reference $TxRef$[11], which encodes details (i.e., chain, block height, transaction index and optionally outpoint index). The transaction can optionally include an OP_RETURN as part of the transaction outputs to refer to a DID document, otherwise, a default document is automatically created. OP_RETURN is a Bitcoin script opcode [2], which can be used to embed up to 80 bytes of data in a transaction. Updating the DID is achieved by spending the current outpoint and setting the OP_RETURN with a reference to the updated DID document. Reading a DID requires a lookup of the $TrxRef$ and following the chain of spending transactions until the last one with an unspent outpoint is reached. If the last transaction has no OP_RETURN, it means the DID has been revoked.

[11] See: https://en.bitcoin.it/wiki/BIP_0136

Rule making *Network and Registry.* For did:btcr, the network and registry are actually the same and correspond to the Bitcoin protocol. Changes to the protocol require drafting a Bitcoin Improvement Proposal (BIP) which is then openly discussed within the community. Therefore, participation in network governance is open and anyone can join, comment and contribute to open debate (open contribution). The process towards BIP acceptance follows the guidelines defined in BIP:2, where it is recommended that the acceptance of a BIP requires at least a 95% acceptance rate by the miners of the last 2016 blocks, unless there is rationale. The deciding group is not closed and includes known and unknown entities/miners (breadth of authority). However, participation in governance cannot be considered fair, as miners with higher hash power have more influence in the decision making. In DID:btcr, although miners receive block rewards and transaction fees, the governance of the DID method itself is decentralized and therefore, established to the public good.
Specification. The specification of the DID:btcr is created and maintained by a closed set of contributors. Comments and suggestions to the specification are open, but the decision lies within a closed and known set of people. Although participation in the specification governance requires time and effort, no incentives to the specification governing entity are defined, thus confirming prior conclusions on financial goals of the method, which is established for the public good without extracting rents or remunerations.

Operation. As the Bitcoin blockchain is public and permissionless, where anyone can participate, read and write (transact) with the ledger, the operations of the did:btcr also do not require any permission. However, resolving a DID without relying on authoritative intermediaries requires operating a full node, which can prevent the use of resource constrained edge devices to directly resolve DIDs. For registering DIDs edge devices can be sufficient unless a continuation document is specified, which would require additional resources for hosting it (e.g., a server). Note that did:btcr supports the creation of both universal and paired DIDs. Although Bitcoin is public and, in principle, anyone can resolve a DID, it is also possible to create a default DID (without setting the OP_RETURN), which would make it indistinguishable from a normal transactions, and therefore can be used in a pairwise manner. It is possible for anyone to retrieve a cryptographic proof of the history of changes (transactions), thus theoretically enabling public auditability. However, referencing continuation documents that reside on mutable storage can hinder these benefits.

Security While relying on Bitcoin transactions as DIDs ensures integrity and persistence, referring to continuation documents raises concerns over censorship and mutability. It is possible for both the storage provider or a governing entity to censor the access to the server hosting the continuation document, rendering the resolution of the corresponding DID impossible without first updating it. Besides censorship, failure of the host server directly impacts the availability of the DID (for resolution) unless a caching mechanism is implemented. Integrity of a continuation document can be checked as it is signed with the transaction input key. However, a discrepancy between the specification document and the

design decision document currendly renders it unclear which input key to use for signing the continuation documents, i.e., (i) the DID creation input key, or (ii) keys used for updates. The latter case would open up the possibility of claiming someone else's DID under certain conditions [4].

Implementation One challenge with implementing btcr DIDs is that there is no single definitive, complete, and up-to-date specification. Implementers have to combine information from different sources, ask questions from the community, and analyze existing examples and code to build a compatible implementation. CRUD operations on btcr DIDs can be difficult when a continuation DID document is needed, since this requires access to a web server, in addition to access to the Bitcoin blockchain. For resolving btcr DIDs, the main challenge is "following the tip", i.e, the process of looking up unspent outpoints of a transaction after a btcr DID has been updated or deactivated, which is not readily supported by all Bitcoin implementations. Finally, an aspect of btcr DIDs is that during creation, the actual DID only becomes fully known and stable after a transaction is mined in a block and sufficiently confirmed. This requires implementations to maintain some kind of internal state and monitoring process.

4.2 did:sov

Description Sovrin is a private non-profit foundation and its DID method relies on a public permissioned blockchain specifically and exclusively targeted for self-sovereign identity [10]. Sovrin's technical underpinnings derive from Hyperledger's Indy project and it employs the plenum protocol, which is an enhancement of RBFT (Redundant Byzantine Fault Tolerance) [1].

Rule Making *Network and Registry.* In did:sov, the network and the registry both correspond to the Sovrin network. Governance is restricted to a board of trustees (BoT) that decide on (i) how the network evolves, and (ii) approval of new stewards or governance proposals by the Sovrin governance framework working group (SGFWG). Stewards are independent entities (e.g., universities, organizations) responsible for endorsing transactions and writing to the ledger, and have to comply with the governance framework approved by the BoT. All governance documents are open to public review and comment, but the decision is ultimately restricted to a closed group (i.e., the BoT). Similar to DNS governance, despite being a non-profit public organization, the Sovrin foundation collects fees and rents to ensure economic viability of the infrastructure. Additionally, although there exist several and different governance bodies within the Sovrin foundation (e.g., SGFWG, STGB, EAC) and any one can join, the ultimate governance approval remains under the BoT control. To be part of the BoT, one should first be nominated by the nomination/transition committee and then voted by the current BoT. Participating in governance is clearly restricted and requires modest costs in terms of efforts and time.
Specification. The specification is governed by the Sovrin technical governance board through the Sovrin trust framework, and revisions (called controlled documents) should undergo the BoT approval.

Operation In contrast to did:btcr, while anyone can read from the Sovrin ledger, writing to it is permissioned and restricted to transaction endorsers (e.g., stewards). Sovrin publicly shares their annual financial reports, which can be checked by anyone, thus rendering its financial accountability transparent. To resolve DIDs, did:sov does not require implementing a full node, but can instead rely on state proofs making the use of edge devices with limited resources possible. However, registering a DID without relying on intermediaries is not possible as it has to be achieved through an endorser. Finally, the public nature of Sovrin enables anyone to retrieve a cryptographic proof of all changes to a DID document, making the system auditable.

Security In practice, did:sov is censorship resistant as it relies on a distributed network of nodes (stewards) from all over the world, responsible for accessing and writing to the ledger. As such, a user that is censored by a steward can readily register a DID using a different one. However, this does not prevent the BoT from issuing new endorsement policies that censor specific type of users, with which endorsers have to comply. The integrity of the ledger is maintained by the diverse stewards and observer nodes, and can also be publicly verified using for example anchored state proofs. Although confidentiality is not a required property, it is possible in did:sov to create pairwise DIDs that are not stored on the ledger. Besides, Sovrin also supports the creation of blinded DIDs using zero knowledge proofs. According to the Sovrin GDPR compliance policies, personal data may not be written to the ledger (data minimization), i.e., only public DIDs and the corresponding DID documents, credential definitions and revocation registries are stored on the ledger. Sovrin also uses software agents (e.g., edge agent, cloud agent) to store and manage credentials and keys, and communicate with other agents in a peer to peer fashion using the didcomm protocol.

Implementation This method was designed for Sovrin, which is widely known and has been used by many implementers and real-life projects. It has also implicitly been designed for other ledger instances of Hyperledger Indy. In practice however, applications and services building on top of these ledgers use custom identifier and discovery formats instead of actual DIDs and DID documents, which has made this DID method hard to understand. A DID method specification exists, but it is outdated. Implementations of sov DIDs therefore are currently mostly based on community knowledge and undocumented assumptions. The ledger itself offers basic operations such as NYM and ATTRIB that make it possible to build DIDs and DID documents on top. The shortcomings and confusion around implementing sov DIDs is expected to be solved with the arrival of the new Indy DID method.

4.3 did:ethr

Description The did:ethr method is similar to did:btcr in that it also builds upon on a blockchain technology, in this case Ethereum. However, while Bitcoin employs a UTXO-based ledger design, Ethereum utilizes an account-based model

and supports quasi-Turing complete *smart contracts* [7]. did:ethr leverages these properties by mapping Ethereum (externally owned) account addresses, which are derived from the public key of an ECDSA Secp256k1 asymmetric key pair, to identities. An important design decision of did:ethr is that the creation of a DID does not necessitate submitting a transaction to the Ethereum network. Instead, any regularly generated externally owned account address is considered a DID. In addition, Ethereum's smart contract functionality is used to realize a registry for CRUD operations and to allow for the delegation of control over an identity. Hereby, the smart-contract-based registry follows the preliminary ERC-1056 standard defined in Ethereum improvement proposal EIP-1056[12] and inherits desirable properties such as immutability, trustless execution and decentralization, from the underlying platform. While did:ethr appears to offer a lightweight and cost effective method for creating DIDs, the design choices introduce some unclear properties regarding the creation and revocation of DIDs that are never committed to through a transaction on the blockchain.

Rule Making *Network and Registry.* Similarly to did:btcr, the network and registry in this method are also provided through the underlying blockchain system. However, did:ethr relies on a registry that is governed by a smart contract which can publicly be interacted with through any Ethereum account or other smart contract code. In regard to rule making, the current smart-contract-based registry specification renders the functionality *immutable* as by the properties of the underlying ledger. However, it is unclear if the current draft ERC-1056 registry design will change to include some ability for governance once the draft is finalized. For the basic network (blockchain), while anyone can participate fully in principle, to be able to meaningfully partake in Ethereum's consensus protocol requires significant hashrate and therefore financial resources. In practice high-hashrate proof of work blockchains such as Bitcoin and Ethereum present themselves more like networks where consensus is permissioned, as the average user has no realistic chance of influencing consensus decisions. In Ethereum, the governance authority is an open set of multiple parties and the process, in analogy to Bitcoin's BIPs, is governed through Ethereum Improvement Proposals (EIPs). The operational costs of the registry are fully transparent because they are publicly visible on the blockchain, and the network costs such as mining rewards and hashrate can also be deduced from on-chain data.

Specification. The did:ethr specification was initially created by uPort [9], however it is not use-case specific or geared toward an extraction of rent and established as a public good. It is openly available in GitHub and is currently governed by the Decentralized Identity Foundation, however there is no reason to assume that this excludes others from actively participating. The smart contract code of the registry is also publicly available and can be checked against the deployed contract.

[12] https://eips.ethereum.org/EIPS/eip-1056

Operation The creation of ethr DIDs does not require any permission, special hardware requirements or even access to the full blockchain ledger, in particular if no transaction to the registry (e.g. for delegating control) is required. For some of the CRUD it can be necessary to interact with the on-chain smart contract through transactions that need to pay transaction fees to miners. These transaction costs are public and recorded in the blockchain. Overall the compensation to miners in Ethereum is highly transparent as all on-chain flows of cryptocurrency funds are publicly accessible. Reading the registry is possible for anyone (either through a light or full node or a third party service). The scheme is designed to be operational on different EVM compatible networks (e.g. Ethereum testnets, Rootstock etc.) and also allows to specify alternative registry addresses. Analogous to did:btcr, if all of the CRUD operations of the DID are performed through transactions, it is possible for anyone to retrieve cryptographic proof of these changes, enabling public auditability. On the other hand, if DIDs are not added to the registry through transactions, they can be used in a pairwise manner and also may offer some degree of privacy e.g., against metadata collection.

Security The advantage of did:ethr lies in the design of not having to commit to the DID in a transaction unless the owner desires so or wants to include properties such as delegating control of the DID to another address. This however means that a DID that has been deleted/revoked which has never used the registry can not be distinguished from one that is not revoked. Integrity is ensured through blockchain consensus and the use of established algorithms for asymmetric cryptography and hash functions. Users can create their DID trustlessly by generating a Secp256k1 keypair. Utilization of the registry is trustless as long as the underlying ledger remains "permissionless", i.e. transactions are not censored by miners and are economically viable. Updates to the DID in the registry are publicly visible and in principle personally identifying information could also be encoded in the DID entry, introducing potential privacy issues. DID resolution and reading the registry can be done with good confidentiality as blockchain state is publicly and anonymously accessible, either through running one's own full or light Ethereum node or, with more trust assumptions, through the API of a third party provider. Similarly to the continuation documents in did:btcr, the method allows for "service endpoints" which can reference external, mutable resources such as URLs, thereby opening up potential concerns over censorship, persistence and privacy.

Implementation This method can render it cost effective for anyone to create DIDs, as the creation step only requires the generation of a cryptographic key pair, without having to perform blockchain transactions. Resolving ethr DIDs requires read operations against Ethereum or the respective Network in which the registry smart contract is located, which can however readily be achieved with any standard Ethereum tools and only incurs modest resource requirements. Once update and deactivate operations are necessitated, implementers need to be able to write to the blockchain, which requires appropriate infrastructure to be in place (similar to other blockchain-based DID methods) and is subject to transaction fees.

4.4 did:web

did:web is a method that uses domain names as identifiers. The DID is a URI that points to a DID document stored on a web host server and registered within a DNS registrar. To resolve the DID, a HTTP GET request on the HTTPS URL generated from the DID is required. Updating the DID is achieved by replacing/updating the DID document on the hosting location. Revocation occurs if the DID document is deleted from the web host.

Rule Making *Network and Registry.* Because the did:web method uses domain names to represent DIDs, both the DID network and registry correspond to the registries and registrars running the domain name servers. Therefore, evaluating governance aspects requires a thorough understanding on how such a traditional DNS system is governed. Governance of DNS is mainly the responsibility of the ICANN, a multistakeholder, private, non-profit organization that follows a bottom-up, consensus driven, model to coordinate the assignment of internet domain names and IP addresses [5]. Although each DNS registry/registrar might have separate internal rules and is responsible for allocating/selling its corresponding domain names, ultimately they have to comply and fulfill the agreements and policies of ICANN. The DNS model combines both public and private economies. The ICANN itself is a non-profit organization that acts for the common good of the public, but extracts registration fees from registries, registrars and indirectly registrants, for covering the running costs of the organization. However, from the perspective of registry and registrars, they extract rents to enhance their profits.

Specification. The specification is published by the credentials community group. Anyone can comment and raise issues through the specification GitHub repository, however, decision making is not clear and seems to be conducted by the specification authors.

Operation While anyone can read, writing to the registry/DNS server is permissioned. Creating a DID requires a subscription within a registrar/registry or a third party seller, in addition to a web host for storing the did document (except the case where users host their own web servers). As such, to resolve a DID using the did:web method without relying on intermediaries, a user has to be an accredited registry. Besides the fact that this does not seem as a practical solution, it will also require exceptional resources and has to follow and comply with complex procedures. Furthermore, while domain names are meant to be used universally (unless using local/private network and DNS server), DID documents are stored on web hosts with no means of cryptographically proving the history of their changes, thus rendering auditability almost impossible.

Security In did:web, censorship can happen at different levels: (i) the DNS or (ii) the web host. While a migration to a new web host would solve the latter scenario (against migration and update fees), the registry has still full control on removing or censoring specific DIDs. Although it is also possible to

transfer the DID to another registrar, the registry (e.g., Verisign for .com) still has the power to deny the user request. In the current specification of did:web, it is not clear how integrity is addressed although a proposal to use hashlinks is suggested. Confidentiality, on the other hand, depends on whether or not the registration within the registrar is private. If not, the user has to reveal her basic information, thus giving registrars the ability to correlate the actual identity with the corresponding DID.

Implementation Resolving web DIDs only requires a simple HTTP GET operation, which can be readily achieved in any programming language or operating system. Creating, updating, and deactivating only require storing and updating a file on a web server.

4.5 did:v1

Description Veres One[13] is a project specifically targeted at the creation and management of DIDs. The v1 specification was drafted by members of Digital Bazaar and is hosted/maintained by the W3C Credentials Community Group. It relies on custom distributed ledger technology, which is based on the "Continuity"[14] BFT consensus protocol that appears specifically targeted for an application in Veres One. The Method is designed to extract rents and remuneration for its operators and it specifies a detailed governance structure for defining the relevant operational entities, governing bodies and how the method specification may be updated. At the time of writing, the collection of specification and design details regarding did:v1 presented itself challenging, as the documentation and code is spread over multiple GitHub repositories and websites and does not paint a coherent picture. Further, while the project management, governance and its goals are outlined, the presented structure is relatively complex and it is unclear how to readily verify if the project adheres to the specification and its claimed goals in practice.

Rule Making *Network and Registry.* v1 intends to use a public ledger where, in principle, anyone can create and resolve DIDs. To keep in line with GDPR compliance, some elements of the DID can exist off chain. While it is claimed that the network is permissionless, operational details suggest that this property may only extend toward the ability of reading the ledger. Specifically, the employed novel consensus protocol and its ability to support an open participation model has not yet received sufficient peer-review to allow for an objective evaluation. The cost for participating as a network node in the Veres One network is not fully clear, but there will be at least a modest cost involved.

Specification. It appears that interested parties can contribute and participate, either by taking on a governance role, or commenting on the public GitHub repositories. However, ultimately control over the specification is held by the Veres One governing body and the entities controlling these repositories.

[13] Cf. https://veres.one/
[14] Cf. https://github.com/digitalbazaar/bedrock-ledger-consensus-continuity

Operation The software necessary to run a client is open source, requires minimal resources and can query information from network nodes. For trustless DID resolution users would have to run a network node themselves. In regard to creating or updating DIDs (writing to the ledger) did:v1 follows a model where users pay an *accelerator fee* that is distributed among the maintainers and participants of the network. Hence, did:v1 offers a more restricted permissioned model similar to did:sov. However, it appears to be possible to circumvent accelerators by performing a proof-of-work or partaking in the protocol as a consensus node.

Security Within did:v1 it is currently unclear if the method can achieve its stated properties in a fully"permissionless" setting in practice, as the consensus protocol is not yet sufficiently analyzed. On the one hand, under the assumption that the ledger and its consensus protocol is fully permissionless, it can achieve censorship resistance. Public verifiability is also possible, however the method also supports external resources which may not be verifiable or could be censored. On the other hand, if consensus is only achieved by assuming a restricted set of participants, i.e., it is permissioned, it opens up the possibility of censorship. According to the method specification, GDPR compliance is achieved but it is not fully clear how this property is enforced in practice. More specifically, to be fully GDPR compliant consensus nodes need to verify that no personally identifiable data has been encoded in the DID, be it intentional or unintentionally. In relation to the right to be forgottenit may be necessary to delete entries in the blockchain's history. However, secure redactable blockchains in the permissionless setting are still a subject of ongoing research [3].

Implementation The did:v1 method directly builds on JSON-LD and Linked Data Proofs, which can provide some familiarity for implementers. Resolving v1 DIDs is straightforward, since each network node exposes an HTTP GET interface for retrieving a fully compliant DID document. One primary question that remains open is the future evolution and implementation of the Veres One ledger. This includes both, whether the envisioned technological goals that are laid out in the specification can be achieved in practice, as well as how governance and network participation (e.g., permissionless or permissioned) is then realized depending on these technologies.

4.6 did:peer

Description The core concept behind the did:peer specification hinges on the insight that there exist two categories of DIDs, namely *anywise DIDs* and *N-wise DIDs*. The former are intended to be used with an unknown number of parties whereas the latter are only intended to be known by exactly N enumerated participants, and the did:peer method addresses this type of DID. A *pairwise DID* is the special case where $N = 2$. N-wise DIDs are only relevant to its corresponding members and aspects such as resolution should only concern the involved parties. Hence the bulk of interactions can be moved off-chain with the possibility of connecting back to a chain-based ecosystem if needed.

Rule Making *Network and Registry.* There are no specific networks for rule making, communications can go through any network channel. The decision on picking a specific network is up to the involved parties participating in did:peer and the registry is only at the peers, held locally. If the peers decide to change network picking and rules or registry, it is up to them. The creator of the DID is only one peer or a pair of peers which agrees on some rules, everything is peer-related. As the network and registry are created between a set of peers, it is only for the common good of those participants.

Specification. The did:peer specification is openly available on Github where anyone can propose improvements or changes, however the board of contributors who can accept such changes seems to be a closed group. It does not appear that the specification is geared toward the extraction of rent and is for the public good.

Operation Anyone can, in principle, participate if she is a peer. However, the network or communication layer is visible only to peers participating in the operations if not otherwise decided by the involved peers. The registry is established between communicating peers and held locally, requiring little overhead or unnecessary data. The network and registry can be anything on what peers agree upon. and DIDs can be created and used contextually, between any set of parties. Auditability of operations depends on the concrete capabilities of the underlying registry and network that was agreed upon.

Security The corresponding DIDs in did:peer are generated in a securely random process. This prevents attackers from discovering patterns in peer DIDs that might undermine privacy. Normally, peer DIDs are not persisted in any central system, so there is no trove to protect. However, in communication with dynamic peers, there is a special layered mechanism which is used to persist others' peer DID docs into backing storage which can be a ledger. Messages in this protocol are sent encrypted, by the specified format DIDComm's encryption envelope. This gives strong guarantees about the confidentiality and integrity of exchanged data. As the communication is mainly between two peers the needed security measures are partially minimized from the network point of view.

Implementation Implementing peer DIDs takes some significant effort for implementers, since this DID method introduces a lot of new concepts that many developers will not be familiar with. On the other hand, the DID method renders it possible to progressively implement more features. Creation of a peer DID only requires generating a key pair, while other operations work via a peer-to-peer protocol between agents. It is not completely clear how peer DIDs currently fit in with other community developments, such as Hyperledger Aries and DIDComm.

5 Discussion and Conclusion

Blockchain technologies have opened up manifold opportunities toward realizing SSI systems that do not need to rely on centralized entities. While, in part, this is achieved through the intrinsic properties of these technologies, e.g., immutability, resistance to censorship, and decentralization, the degree to which

a DID method can be considered decentralized or secure also depends on many other aspects and design choices. One cannot assume that just because a DID method is based on blockchain technology implicitly renders it decentralized. Indeed, it is extremely important to consider all dimensions, i.e., (i) network, (ii) registry, and (iii) specification, and assess a method's fundamental properties (e.g., governance, economic model or security) against each of these dimensions. Table 1 provides a comparative overview of the investigated methods. While *protection* determines a method' resistance to censorship, *persistence* evaluates the longevity of decentralized identifiers. *Integrity* ensures that DiDs and the corresponding DiD documents have not been tampered with, and *confidentiality* means that the DiD method gives the option to protect DiDs or DiD Documents from unauthorized disclosure if required. Finally, *Decentralization* examines how decentralized is a DiD method by evaluating the decentralization of its underlying network, registry and specification governance, and operations. For example, our evaluation has shown that despite did:btcr relying on the decentralized Bitcoin network for creating DIDs, the corresponding DID documents are still hosted on mutable storage, thus hindering blockchain benefits and introducing new risks of censorship, availability and persistence. Similarly, while did:ethr also builds upon a public permissionless Blockchain (Ethereum), its design relies on an on-chain ERC-1056 smart contract to manage and govern the DID registry. By doing so, trust is shifted to both, the smart contract implementation as well as the underlying Ethereum blockchain. Note, that in both methods changes to the specification cannot prevent that operations on DIDs can also follow previous specification versions. Resolving them would hence require DID resolvers to maintain all previous resolution implementation versions. It is noteworthy to point out that most of the evaluated methods have a distinct lack of version control/migration mechanisms to prevent old DIDs from becoming unresolvable or obsolete. A clear definition on how to upgrade the method specification and assessment of its impact on the current implementation would clearly prove beneficial for introducing new features and mitigating security or performance issues.

There exist security, usability and scalability trade-offs between methods employing identity-specific ledgers and methods relying on public blockchains, and between the permissioned and permissionless operation of the underlying ledger. Identity-specific permissioned ledgers that rely on BFT-based consensus mechanism (e.g., Sovrin and Veres One) offer the advantage of better scalability and performance over traditional Blockchain designs (e.g., Bitcoin and Ethereum), however this comes at the cost of reduced decentralization and an increased risk of censorship by operators. This derives from the fact that the entities responsible for writing to the ledger have to comply with endorsement agreements that, in the end, might be changed by the governing entity, which to a certain extent is less decentralized and may serve specific interests.

The selected methods interestingly rely on different economic models that range from non-profit organizations which extract rents from the DID methods to totally open and free (not considering network fees) community projects. This

Table 1. Comparative table of DID methods

	RuleMaking			Operation		Security*			
	Network	Registry	Specification	Network	Registry	Pro	Per	Int	Conf
did:btcr	● □	● □	○	● □	● □	+	+	+	±
did:v1	◑†	◑†	○	◑†	◑†	−	±	±	+†
did:ethr	● □	$N/A^{\dagger}$	◑	● □	● ■	+	+	+	−
did:sov	◑ ■	◑ ■	◑	◑ ■	◑ ■	−	±	+	±
did:web	◑	◑	○	◑ □	◑ □	−	−	−	±
did:peer	●	●	○	◑†	◑†	±	−	−	+

***** Security - Pro: protection Per: persistence Int: integrity Conf: confidentiality

● fully decentralized ◑ partially decentralized ○ centralized $N/A^{\dagger}$ not applicable

Required resources: ■ modest □ substantial

† Not clear or well defined how method satisfies criteria at time of writing

creates a trade-off between the sustainability and growth of the project and trust in the system. Indeed, participation in governance and maintenance often requires substantial efforts and time. Hence, a failure to consider aligning incentives or covering operational costs in the method's economic model may lead to a stale project or to an outdated specification. On the other hand, incentives should not be aligned to only serve the economic interest of a specific entity, thus diminishing trust, openness, transparency and accountability of the system.

We hereby point out some of the challenges encountered while conducting this evaluation. First, the amount and quality of available documentation, as well as the discrepancy between some of the methods' specifications and their actual corresponding implementations, introduced uncertainties on how to fairly evaluate specific properties of the method. Moreover, some of the specifications might have changed during and after the evaluation, thus requiring continuous revision of the evaluation. Finally, some of the constructs and goals of specific methods are difficult to verify in practice or have yet to be implemented, leaving an answer to whether or not they can achieve the promised guarantees unclear.

To conclude, there is no clear winner among the evaluated DID methods. Each comes with advantages and disadvantages, and the selection of a particular method heavily depends upon the use case (e.g., supply chain, KYC, automotive process) and desired properties. Some application areas may require scalable and private systems, while others can necessitate a focus on distribution and trust. Furthermore, while blockchain offers unique security and decentralization properties for DID methods, it does not prevent flawed specifications and governance designs from introducing vulnerabilities that could jeopardize potential benefits.

Acknowledgments. This research is based upon work partially supported by (1) the Christian-Doppler-Laboratory for Security and Quality Improvement in the Production System Lifecycle; The financial support by the Austrian Federal Ministry for Digital and Economic Affairs, the Nation Foundation for Research, Technology and Development and University of Vienna, Faculty of Computer Science, Security & Privacy Group is gratefully acknowledged; (2) SBA Research (SBA-K1); SBA Research is a COMET Center within the COMET – Competence Centers for Excellent Technologies Programme and funded by BMK, BMDW, and the federal state of Vienna. The COMET Programme is managed by FFG. (3) the FFG ICT of the Future project 874019 dIdentity & dApps. (4) the FFG Industrial PhD project 878835 SmartDLP. (5) the U.S Department of Homeland Security Science and Technology Directorate's Silicon Valley Innovation Program (SVIP) under OTA 70RSAT20T00000030. Any opinions contained herein are those of the author(s) and do not necessarily reflect those of DHS S&T

References

1. Aublin, P.L., Mokhtar, S.B., Quéma, V.: Rbft: Redundant byzantine fault tolerance. In: 2013 IEEE 33rd International Conference on Distributed Computing Systems. pp. 297–306. IEEE (2013)
2. Bartoletti, M., Pompianu, L.: An analysis of bitcoin op_return metadata. In: International Conference on Financial Cryptography and Data Security. pp. 218–230 (2017)
3. Deuber, D., Magri, B., Thyagarajan, S.A.K.: Redactable blockchain in the permissionless setting. In: 2019 IEEE Symposium on Security and Privacy (SP). pp. 124–138. IEEE (2019)
4. Fdhila, W., Stifter, N., Kostal, K., Saglam, C., Sabadello, M.: DID methods evaluation report - draft (January 2021), https://docs.google.com/document/d/1jP-76ul0FZ3H8dChqT2hMtlzvL6B3famQbseZQ0AGS8/
5. Klein, H.: Icann and internet governance: Leveraging technical coordination to realize global public policy. The Information Society **18**(3), 193–207 (2002)
6. Lesavre, L., Varin, P., Mell, P., Davidson, M., Shook, J.: A taxonomic approach to understanding emerging blockchain identity management systems. CoRR **abs/1807.06346** (2019)
7. Luu, L., Chu, D.H., Olickel, H., Saxena, P., Hobor, A.: Making smart contracts smarter. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. pp. 254–269 (2016)
8. Mühle, A., Grüner, A., Gayvoronskaya, T., Meinel, C.: A survey on essential components of a self-sovereign identity. CoRR **abs/1807.06346** (2018)
9. Naik, N., Jenkins, P.: uport open-source identity management system: An assessment of self-sovereign identity and user-centric data platform built on blockchain. In: 2020 IEEE International Symposium on Systems Engineering. pp. 1–7 (2020)
10. Reed, D., Law, J., Hardman, D.: The technical foundations of sovrin. Technical report, Sovrin, 2016 (2016)
11. Stifter, N., Judmayer, A., Weippl, E.: Revisiting practical byzantine fault tolerance through blockchain technologies. In: Security and Quality in Cyber-Physical Systems Engineering, pp. 471–495. Springer (2019)