Towards a Blockchain-based Identity Provider

Andreas Grüner, Alexander Mühle, Tatiana Gayvoronskaya, Christoph Meinel
Hasso Plattner Institute (HPI)
University of Potsdam, 14482, Potsdam, Germany
Email: {andreas.gruener, alexander.muehle, tatiana.gayvoronskaya, christoph.meinel}@hpi.uni-potsdam.de

***Abstract*—The emerging technology blockchain is under way to revolutionize various fields. One significant domain to apply blockchain is identity management. In traditional identity management, a centralized identity provider, representing a trusted third party, supplies digital identities and their attributes. The identity provider controls and owns digital identities instead of the associated subjects and therefore, constitutes a single point of failure and compromise. To overcome the need for this trusted third party, blockchain enables the creation of a decentralized identity provider serving digital identities that are under full control of the associated subject. In this paper, we outline the design and implementation of a decentralized identity provider using an unpermissioned blockchain. Digital identities are partially stored on the blockchain and their attributes are modelled as verifiable claims, consisting of claims and attestations. In addition to that, the identity provider implements the OpenID Connect protocol to promote seamless integration into existing application landscapes. We provide a sample authentication workflow for a user at an online shop to show practical feasibility.**

Keywords–Blockchain, distributed ledger technology, digital identity, self-sovereign identity, Ethereum

I. Introduction

In 2008, Satoshi Nakamoto published the foundational paper on Bitcoin and started the rise of its underlying blockchain technology [1]. Bitcoin is the first popular digital currency based on a peer-to-peer network without the involvement of a trusted third party. The concept of a decentralized digital currency scheme is generalized by the decentralized execution of additional computations. Bitcoin provides a limited scripting language to enforce requirements on the processing of payments [1]. Beyond this, the Ethereum blockchain comprises a Turing-complete virtual machine for the execution of arbitrary code [2]. This capability allows the implementation of smart contracts [3] to specify complex behaviour for payments or value transfer in general. On top of that, it enables further applications without requiring a centralized entity. Thus, current blockchain technology allows decentralized storage and execution of applications within a network of peers, eliminating the need for a trusted third party [4].

Identity management is concerned with the representation and administration of entities and their attributes as digital identities. Digital identities serve in the identification, authentication and authorization process for applications [5]. The security of an application significantly depends on recognizing users and preventing impersonation attacks of other users. In this regard, secure identification and authentication procedures are fundamental to avoid misuse.Furthermore, authorization ensures that properly authenticated users act within granted privileges. Therefore, identity management is a substantial cornerstone in securing the digital world and in preventing fraud.

A pivotal entity in this domain is an identity provider. The identity provider implements identification, authentication and authorization functions and provides these services to other parties [6]. Traditionally, an identity provider represents a trusted third party and is used within an organization. In addition to that, identity providers that are external to organizations are used in identity federation scenarios. An end user wants to authenticate at a service provider. The service provider redirects the end user to the identity provider for this process. The identity provider confirms a successful login or reports a failed authentication to the service provider. Based on the result, access to the offered service is granted or denied.

A service provider significantly relies on the proper execution of the processes carried out by the identity provider. This trust is mainly derived from contractual obligations, due diligence and reputation of the identity provider. Overall, in traditional identity management, the identity provider is a trusted third party and essential to the security of applications.

The centralized identity provider as the trusted third party has several downsides. First and foremost, the identity provider needs to be trusted due to centralized control and ownership of digital identities and their attributes. The subject of the digital identity is not in possession of its own data. Additionally, the identity provider represents a single point of failure and therefore decreased reliability. As a central entity the identity provider may accumulate a large amount of identity data and becomes a profitable target to attackers, thereby increasing motivation for data theft.

To address these challenges, we have devised a decentralized implementation of an identity provider using an unpermissioned blockchain. The blockchain-based identity provider removes the trusted third party from identity management and remediates centralized control and ownership of the digital identities as well as the single point of failure and compromise. Trust in the decenrally issued identities is derived from the transparency of the blockchain implementation and the attestation issuers, that verify claims. Additionally, the OpenID Connect [7] protocol is implemented to facilitate seamless integration into existing application landscapes and eases the transition from conventional providers.

The remainder of this paper is structured in the following way. In Section 2, we present related work and concepts. The subsequent section provides background on the interrelations between blockchain technology and identity management. We devise our blockchain-based identity provider in Section 4. Section 5 describes a sample authentication workflow using the implemented identity provider. We provide suggestions for future work in Section 6 and conclude the paper in Section 7.

II. Related Work

Numerous practical and academic projects combine blockchain technology and identity management [8]. These projects target either specific parts of identity management or are directly concerned with a self-sovereign identity. Implementation approaches differ between creating specific-purpose blockchains or adding functionality on top of existing blockchains using smart contracts. However, the majority of projects offer only a limited amount of detail regarding the technical implementation. In the following section, we describe uPort and Sovrin due to the sufficient amount of available information and the maturity of the solutions. Additionally, we point out differences to our blockchain-based identity provider.
A comprehensive self-sovereign identity solution is implemented by uPort [9] in the form of smart contracts on the Ethereum blockchain. A digital identity is mainly represented as a controller, proxy, and recovery contract. The address of a proxy contract is the identifier of the digital identity. The controller contract establishes a management function to administrate and use the proxy contract as an identity. This distinction enables the replacement of the controller contract and fosters persistence of the proxy contract address. The restoration of the private key is the intent of the recovery contract. Additionally, a central and user-independent registry contract on the blockchain is used to reflect bindings between identities and claims or attestations. Claims and attestations are stored on InterPlanetary File System (IPFS) [10] or central cloud storages. Besides blockchain-based components of uPort, there are additional elements of the ecosystem. A developer library enables the integration into applications. The uPort mobile app is the key application for the end user to manage the digital identity.
Compared to uPort, our blockchain-based identity provider solution is implemented as dedicated unpermissioned blockchain yielding a benefit on computational efficiency and reduced transaction cost. uPort uses the general execution environment and transaction costs on Ethereum. Our identity provider is directly integrated into a blockchain and uses dedicated transactions. Besides that, our identity provider offers OpenID Connect conformity to seamlessly integrate into existing application landscapes.
Sovrin [11] is a public and permissioned blockchain solution dedicated to providing identity management. Sovrin nodes are distinguished as validator or observer nodes. Validators are specifically chosen nodes that are permissioned to write the next state of the blockchain and include transactions. Observer nodes solely read the blockchain and make the information available for clients. Sovrin is supervised by a complex trust framework with different governance bodies that make decisions on the further development of the blockchain and the admission of new validator nodes. Additionally, participation in the network is liable to contractual agreements issued by the Sovrin Foundation [12]. A digital identity of Sovrin comprises an identifier and attributes are modelled as claims and attestations. Aliases can be linked to the identifier to increase privacy. Several claim types are differentiated that enable, for instance, clear, encrypted and hashed storage on the blockchain. Storage providers can be used to save the data in case the claim is not directly stored on the blockchain.
In contrast to Sovrin and the use of governance bodies, our blockchain-based identity provider utilizes an unpermissioned blockchain to avoid reliance on trusted third parties and to foster the vision of a self-sovereign identity.

III. Blockchain and Identity Management

Considering both domains, blockchain technology and identity management, there is mutual interest and applicability. On the one hand, a permissioned blockchain requires the implementation of identity management and access control to grant privileges on the blockchain layer to eligible participants. A permissioned blockchain comprises predetermined nodes for transaction processing and block creation [13]. The predetermined nodes need to be identified and permissions must be assigned to the respective digital identities.
On the other hand, using blockchain technology to build a distributed execution environment for self-sovereign identities forms a distinct identity provider. Blockchain technology enables the implementation of a decentralized digital identity that is not issued and owned by a trusted third party. This digital identity is under true control of its associated entity. Therefore, a decentralized digital identity adhering to specific characteristics is named a self-sovereign identity. These properties are elaborated by Allen [14] and can be grouped into the categories security, controllability and portability [15]. The cluster security comprises protection, minimisation and persistence. Protection refers to the general precedence of the digital identity's owner rights. Minimisation is concerned with data privacy and the reduction of information disclosure about the subject. Persistence describes the long-term existence of a digital identity. Controllability is the second category in the attribute grouping and encompasses existence, control and consent. Furthermore, persistence is repeatedly indicated. Existence describes, that a digital identity should reflect a physical object. The control of the identity is completely in the possession of the owner and without the consent of the owner no information is revealed. Portability is the last category and comprises interoperability, transparency and access. The digital identity and corresponding identity provider services are interoperable with customers and provider services applying standard protocols. The implementation, operation and actioning of the digital identity is transparent to all involved parties. The owner, or any legitimate party, has easy access to information or attributes of the digital identity. Overall, blockchain technology is able to provide decentralized identity management for other applications in a novel way.

IV. A Blockchain-based Identity Provider

In the following sections, we outline our decentralized identity provider based on blockchain technology. Starting with objectives and requirements that lead to particular design decisions, we subsequently present the overall architecture, theoretic model and implementation of the novel identity provider.

A. Objective

In traditional identity management, digital identities and their attributes are issued by a centralized identity provider that represents a trusted third party. Service providers need to trust the correctness of the identity provider as well as the validity of issued digital identities and their attributes. In addition to that, trust is required in properly performing the authentication process of a subject. Furthermore, the centralized identity provider

is in full control and ownership of the digital identity and its attributes. Therefore, the subject needs to trust the identity provider on carefully handling and protecting its data. Besides that, trust in compliant behaviour according to regulation and contracts of the identity provider is required. The subject does not expect arbitrary actions, for instance revoking attributes or the complete digital identity, leaving the subject without access to potential critical resources. An identity provider usually serves numerous subjects and therefore collects and stores an accumulated amount of data being a profitable target for attackers. Overall, a centralized identity provider represents a single point of compromise and control.
To overcome these challenges, blockchain technology enables the implementation of a decentralized identity provider without it being a trusted third party. We devise a novel implementation approach of an identity provider using an unpermissioned blockchain to decentralize identity management and derive trust in digital identities from claims and attestations instead of the identity provider itself. The blockchain-based identity provider applies conventional protocols to seamlessly integrate into existing application landscapes confining required changes on the side of the service provider.

B. Requirements

Besides the general objective, we consider the following requirements as significant for our blockchain-based identity provider.

- Decentralization. Decentralization of the identity provider model is a key factor to foster independence from a central authority. In general, decentralization is enabled by the blockchain model. However, an introduction of concentrated external dependencies needs to be prevented in the blockchain network.
- Standard Protocols. The usage of identity management protocols as standards is necessary to foster a seamless integration and migration from conventional identity providers to the blockchain-based identity provider.
- Efficiency. The identity provider should be cost efficient with regards to transaction fees to foster its usage.

C. Design Decisions

There are different solution approaches to building a blockchain-based identity provider that fulfils the stated objective and implements the listed requirements. We make the subsequent design decisions to achieve an optimal solution.
The identity provider is implemented as a separate blockchain instead of a smart contract-based approach on an existing general purpose blockchain. Using smart contracts on another blockchain affects efficiency in terms of computation and cost. A dedicated identity provider blockchain implements the required components more efficiently compared to an execution on a general purpose distributed virtual machine. Furthermore, relying on a general purpose blockchain implies the adoption of the respective transaction fee cost model. Adjusted transaction costs to identity management yield a cost benefit.
To concentrate on the development of the identity provider, we fork an existing blockchain as the foundation and integrate the identity provider as a core component. We determined Ethereum as the most suitable solution for our identity provider based on the broad community, extensive documentation and published source code of the different clients.
Furthermore, we chose the OpenID Connect protocol as integration pattern into existing applications. OpenID Connect specification as an amendment of OAuth 2.0 [16] is developed by major technology companies and has wide adoption. Besides that, identity federation with social networks (e.g. Facebook) are highly used.

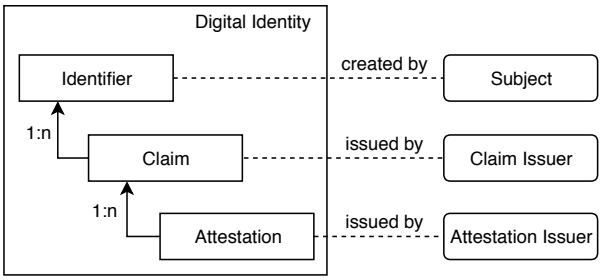

Figure 1. Digital Identity Model and Actors

D. Digital Identity Model and Actors

The digital identity comprises a unique identifier and attributes. The identifier is chosen arbitrarily by the subject upon creation of the identity. Uniqueness is ensured due to recording and verification on the blockchain network. The attributes of the digital identity are modelled as claims and attestations. A claim is a statement about an attribute of the digital identity. The attestation of a claim is an assertion about the correctness and validity of a claim by a digital identity. See Figure 1 for an overview of the model.
The digital identity is created by a subject generally referring to an end user. The claim issuer creates statements about the identity and the attestation issuer asserts these statements. The service provider offers services to end users. To use a service the subject authenticates and potentially authorizes itself to the service provider by using the blockchain-based identity provider. Both end user and service provider can act as claim and attestation issuer.

E. Architecture and Authentication Process

In traditional identity management, the subject, identity provider and service provider represent distinct entities. The subject registers at the identity provider to create a digital identity and potentially provide information about attributes. The service provider forwards the subject to the identity provider during the authentication process. The subject proves with credentials to be in control of the respective digital identity and the identity provider sends the authentication result to the service provider.
Using a blockchain-based identity provider, the distinct entity of an identity provider is replaced by a blockchain network leading to changes in the general architecture and the authentication process. An overview of the architecture is depicted in Figure 2. Subject and service provider each operate a node in the network to establish a connection to the decentralized identity provider. Initially, the subject creates a digital identity by issuing a transaction to the network.
Upon requesting access at a service provider (for instance at an online shop) the service provider forwards the subject to the

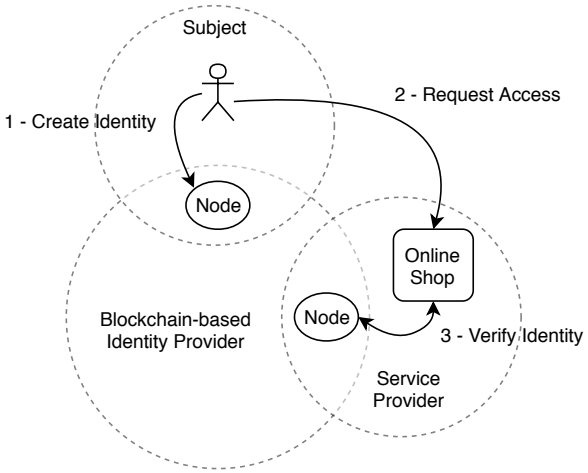

Figure 2. Architecture

local node of the identity provider. Subsequently, the subject proves to be in control of the presented digital identity and the service provider grants access to its portal.

F. Theoretic Model

Our blockchain-based identity provider is based on the Ethereum blockchain. Therefore, we extend the world state of Ethereum as the theoretic model by an additional identity state. The state transition function is modified to embrace changes of the identity state resulting from newly introduced identity transactions that are recognized by the blockchain.

1) World State and Identity State:
The entire state is named world state and comprises address to account state associations [2]. We extend the world state to additionally include mappings from addresses to identity states aligned to the account states and formally define it as follows.

$$\sigma = (A, I)$$

A comprises the account states as defined in [2] with $A_{[m]}$ referencing a specific account by address m. We define I as the set of identity states with $I_{[m]}$ referencing the identity state of address m. An identity state contains the following attributes.

- Nonce n. A scalar value matching the changes of the identity. An identity is created with nonce = 1.
- Identifier i. An arbitrary string that references the digital identity.
- Owner o. Owner represents the related account of the identity. This account controls the digital identity.
- Claims c. The attribute comprises a cryptographic hash of a trie that stores the claims of the identity. The data of a claim might be stored on the blockchain or outside the blockchain network. In case the data of the claim is stored at another storage provider a cryptographic hash is added as information of the claim to the blockchain.
- Attestations a. The property contains a cryptographic hash of a trie that stores attestations for the claims of the digital identity. Comparable to claims, the attestations can be stored on the blockchain or on another storage solution having the cryptographic hash on the blockchain.

The identity state is formally defined as follows.

$$I_{[m]} = (n, i, o, c, a)$$

2) Transactions:
A transaction is a cryptographically signed message to the blockchain network. There are two types of transactions T: Contract creation transaction T_{con} and message call transaction T_{msg} [2]. These transactions are determined to evolve the account state. We introduce three additional transaction types to facilitate the identity model and allow identity state changes. These transactions are as follows.

- Create Identity T_{cre}. An identity is initially created by specifying the identifier i. The owner o is indirectly set to the account from which the transaction originates.
- Modify Identity T_{mod}. An identity is modified during its lifetime by adding or removing claims and attestations.
- Delete Identity T_{del}. An identity is deactivated by removing the owner as well as clearing claims and attestations. Therefore, the control of the identity is revoked and no further actions are possible.

We extend the definition of a transaction T in [2] to comprise the following fields.

- Type p. The attribute specifies the transaction type and is one of T_{con}, T_{msg}, T_{cre}, T_{mod} or T_{del}.
- Nonce n. The nonce determines the count of transactions generated by the sender that is defined with the attribute from f.
- GasPrice p. Gas is consumed for executing computations of the transaction. Gas price p is the cost for one unit of gas.
- GasLimit g. The field determines the upper bound of gas used for the transaction.
- To t. The property defines the recipient of the transaction.
- From f. The field characterizes the originator of the transaction either being an account itself or an identity.
- Value v. Value v defines the payment transferred to the recipient of the transaction.
- Signature w, r, s. The properties comprise the cryptographic signature of the transaction by the sender as defined in [2].
- Init i. Data used for transaction of type T_{con}.
- Data d. Data used for transaction of type T_{msg} and T_{mod}.

The general validity of a transaction is determined through the verification of the sender's cryptographic signature. A valid transaction containing the sender's address of an account is signed with the corresponding key pair. Additional basic transaction verification steps are defined in [2]. Invalid transactions are not processed.

3) State Transition:
The world state transitions into a new state based on transactions issued to the network. These transactions advance the world state's underlying account [2]. Additionally, identity transactions update the identity states. The mining of the next

block of the blockchain persists the included transactions and advertises the state evolvement to all nodes of the network.
The state transition function Υ advances the world state σ to the new world state σ' based on a Transaction T and is formally defined as follows [2].

$$\sigma' = \Upsilon(\sigma, T)$$

We detach account state transitions from identity state transitions and define the following sub functions of Υ.

$$(A, I)' = \Upsilon((A, I), T)$$

$$\Leftrightarrow$$

$$\Upsilon_A(A, T) = \begin{cases} A', & T \in \{T_{con}, T_{msg}\} \\ A, & T \notin \{T_{con}, T_{msg}\} \end{cases}$$

$$\wedge\Upsilon_I(I, T) = \begin{cases} I', & T \in \{T_{cre}, T_{mod}, T_{del}\} \\ I, & T \notin \{T_{cre}, T_{mod}, T_{del}\} \end{cases}$$

The identity state transition function Υ_I is the main function of the blockchain-based identity provider.

G. Implementation

The foundation of our blockchain-based Identity Provider (bbIdP) is the Python client of Ethereum comprising the main libraries pyethapp [17] and pyethereum [18]. We adapted the pyethereum implementation according to the theoretical model to support the newly introduced identity management transactions and to store identity information in a separate identity state. Pyethapp is modified to use the updated pyethereum library accordingly. To fully leverage the identity provider model, pyethapp's service oriented architecture is extended by an OpenID Connect provider based on the pyoidc library [19] to offer respective service and achieve straightforward integration.
The representation of identifier, claims and attestations differentiates an internal and external model. The external model is aligned to standards under development by World Wide Web Consortium (W3C) community working groups [20] [21]. The internal specification is a reduced representation to facilitate a streamlined implementation. The blockchain-based identity provider offers remote procedure calls to retrieve identifier, claims and attestations in the external format. Additionally, the OpenID Connect provider accepts the external representation. The format of the identifier is aligned to the Decentralized IDentifier (DID) specification [20] and defined as a particular DID method scheme (see Figure 3). The method namespace is bbidp and abbreviates the blockchain-based identity provider proposed in this paper. The portion idstring is a combination of one or more characters or numbers. This identifier is specified during the creation of the digital identity. It is provided in the "To" attribute of the identity creation transaction. To externally reference the identity, the fully qualified decentralized identifier is used.
In general, the external structure of a claim follows the credential entity model of the Verifiable Claims [21] community working group. A claim is represented in the JavaScript Object Notation (JSON) [22] format. A sample is shown in Figure 4. Each claim consists of a claim identifier, meta data and a property that contains the actual attribute of the digital identity. A claim is issued in simplified form to the blockchain contained in the data field of the identity modification transaction. Issuer and issue timestamp are implicitly obtained from the respective transaction. The specified attribute of the identity can be issued as a cryptographic hash to increase privacy. Internally, the claim is stored in the claim trie of the appropriate identity. The key is the claim identifier and the value is represented by the remaining attributes. To revoke an existing claim, a transaction is issued containing a claim with the same identifier that has no claim attribute.
The attestation of a claim is a signature of the claim itself by the attestation issuer. It is represented in JSON and internally stored in the attestation trie of the identity. Additionally, the attestation comprises meta data about the issuer, creation time and the referred claim. In contrast to the Verifiable Claims working group, we internally separated the attestation from the claim to allow various attestations of a single claim from different attestation issuers.
The integrated OpenID Connect provider serves a simple web page. Upon re-directing a user from the originating portal for authentication, it provides a random value encoded as Quick Response (QR) code [23]. The provider expects as return value a JSON data structure containing the random value and the identity profile that is signed by the owner account of the digital identity. Subsequently, the provider verifies against the blockchain, that the signature is valid and the used account corresponds to the owner of the digital identity. In case of positive verification, the provider returns a positive message and redirects the user back to the originating portal. In case of authentication failure, an error message is delivered.

```
      did = "did:bbidp:" idstring
  idstring = 1*idchar
    idchar = ALPHA / DIGIT
```

Figure 3. bbIdP DID Method Scheme

```
{
  "id": "identifier"
  "type" : "Smith",
  "claim" : {
    "id" : "did:bbidp:bob"
    "firstname" : Bob
  }
}
```

Figure 4. Sample Claim

V. Sample Workflow

Alice owns an online book shop. To order a book a customer needs to login to the online shop. The online shop offers the possibility to login with our blockchain-based identity provider (see Figure 5).
Bob wants to buy products in Alice's online shop. He creates a digital identity on the blockchain-based identity provider network by issuing an identity creation transaction with the identifier "bob". After selecting products in the online shop, Bob navigates to the sign-in page. Next, the blockchain-based identity provider is chosen as a login method by Bob. Consequently, the identity provider generates a random value and provides it in the form of a QR code to the online shop

in an iFrame. Bob signs the random value and the profile of his digital identity related by the identifier "did:bbidp:bob" and sends it to the embedded callback address of the identity provider. Upon successful verification of the return message Alice's online shop recognises Bob.

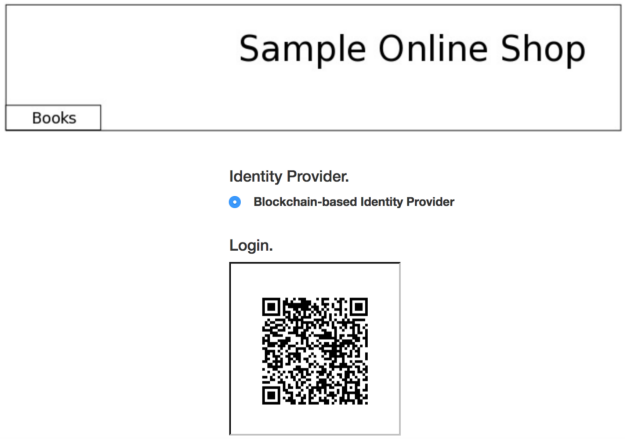

Figure 5. Sample Online Shop

VI. Future Work

A future enhancement for our blockchain-based identity provider is the functional extension to utilize claims and attestations for the purpose of authorization in alignment with the OAuth 2.0 protocol. A service provider could add an attestation of a purchased service to the digital identity of a customer. Based on the attested claim, the service provider can grant access to the purchased offering upon return of the customer to the online service. An additional research area is related to the security of the public unpermissioned blockchain, that is used as an identity provider. Remain the security assumptions for a general purpose blockchain valid in case of a dedicated blockchain for identity management.

VII. Conclusion

Blockchain technology enables the creation of a decentralized identity provider without a trusted third party. We presented the design and implementation of a novel blockchain-based identity provider that offers digital identities containing verifiable claims. The blockchain-based identity provider conforms to the OpenID Connect protocol in order to integrate seamlessly in existing authentication processes. The conjunction of the conventional OpenID Connect protocol with the novel blockchain-based identity provider model enables overarching usage of these technologies. Finally, we described a sample authentication workflow to show practical feasibility.

References

[1] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. [Online]. Available: https://bitcoin.org/bitcoin.pdf [retrieved: 2018-07-18] (2008)

[2] G. Wood. Ethereum: A secure decentralised generalised transaction ledger. [Online]. Available: https://pdfs.semanticscholar.org/ac15/ea808ef3b17ad754f91d3a00fedc8f96b929.pdf [retrieved: 2018-07-18]

[3] N. Szabo. Smart contracts: Building blocks for digital markets. [Online]. Available: http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html [retrieved: 2018-07-18] (1996)

[4] C. Meinel, T. Gayvoronskaya, and M. Schnjakin, "Blockchain: Hype oder innovation," Hasso-Plattner Institute, Prof.-Dr.-Helmert-Strae 2-3, 14482 Potsdam, Germany, 2018.

[5] G. Williamson, D. Yip, I. Sharoni, and K. Spaulding, Identity Management: A Primer. MC Press Online, LP., 2009.

[6] MIT. Information systems & technology website. the knowledge base. idp (identity provider). [Online]. Available: http://kb.mit.edu/confluence/display/glossary/IdP+(Identity+Provider) [retrieved: 2018-07-18]

[7] OpenID Foundation. Openid connect. [Online]. Available: http://openid.net/connect/ [retrieved: 2018-07-18]

[8] Blockchain and identity. [Online]. Available: https://github.com/peacekeeper/blockchain-identity [retrieved: 2018-07-19] (2018)

[9] C. Lundkvist, R. Heck, J. Torstensson, Z. Mitton, and M. Sena. uport: A platform for self-sovereign identity. [Online]. Available: http://blockchainlab.com/pdf/uPort_whitepaper_DRAFT20161020.pdf [retrieved: 2018-07-19] (2016)

[10] J. Benet. Ipfs. content addressed, versioned, p2p file system. [Online]. Available: https://arxiv.org/pdf/1407.3561.pdf [retrieved: 2018-07-19] (2014)

[11] D. Reed, J. Law, and D. Hardman. The technical foundations of sovrin. a white paper from the sovrin foundation. [Online]. Available: https://www.evernym.com/wp-content/uploads/2017/07/The-Technical-Foundations-of-Sovrin.pdf [retrieved: 2018-07-19] (2016)

[12] D. Reed et al. Sovrin provisional trust framework. [Online]. Available: https://sovrin.org/wp-content/uploads/2018/03/Sovrin-Provisional-Trust-Framework-2017-06-28.pdf [Accessed: 2018-07-19] (2017)

[13] BitFury Group. Public versus private blockchains. part 1: Permissioned blockchains. white paper. [Online]. Available: https://bitfury.com/content/downloads/public-vs-private-pt1-1.pdf [retrieved: 2018-07-19] (2015)

[14] C. Allen. The path to self-sovereign identity. [Online]. Available: http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html/cointelegraph.com/news/first-iteration-of-ethereum-metropolis-hard-fork-to-appear-monday [retrieved: 2018-07-18] (2016)

[15] A. Tobin and D. Reed. The inevitable rise of self-sovereign identity. a white paper from the sovrin foundation. [Online]. Available: https://sovrin.org/wp-content/uploads/2017/06/The-Inevitable-Rise-of-Self-Sovereign-Identity.pdf [retrieved: 2017-07-19] (2017)

[16] Internet Engineering Task Force. Request for comments: 6749. the oauth 2.0 authorization framework. [Online]. Available: https://tools.ietf.org/html/rfc6749 [retrieved: 2017-07-19] (2012)

[17] Pyethapp. [Online]. Available: https://github.com/ethereum/pyethapp [retrieved: 2018-07-18]

[18] Pyethereum. [Online]. Available: https://github.com/ethereum/pyethereum [retrieved: 2018-07-19]

[19] Pyoidc. [Online]. Available: https://github.com/OpenIDC/pyoidc [retrieved: 2018-07-16]

[20] D. Reed et al. W3c community group draft report. decentralized identifiers (dids) v0.9. data model and syntaxes for decentralized identifiers (dids). [Online]. Available: https://w3c-ccg.github.io/did-spec/ [retrieved: 2018-07-18] (2018)

[21] M. Sporny and D. Longley. W3c community group draft report. verifiable claims data model and representations 1.0. [Online]. Available: https://www.w3.org/2017/05/vc-data-model/CGFR/2017-05-01/ [retrieved: 2018-06-15] (2018)

[22] Internet Engineering Task Force. Request for comments: 7159. the javascript object notation (json) data interchange format. [Online]. Available: https://tools.ietf.org/html/rfc7159 [retrieved: 2018-07-20] (2014)

[23] International Standardization Organization. Iso/iec 18004:2000. information technology - automatic identification and data capture techniques - bar code symbology - qr code. [Online]. Available: https://tools.ietf.org/html/rfc7159 [retrieved: 2018-07-20] (2000)