Self-Sovereign Identity for IoT Devices

Nataliia Kulabukhova(✉), Andrei Ivashchenko, Iurii Tipikin, and Igor Minin

Saint-Petersburg State University, Saint Petersburg, Russia
n.kulabukhova@spbu.ru

Abstract. This work is an overview of different approaches to the self-sovereign identity (SSI) concept. The idea of constructing of a digital passport for each and every person in the world is not unique, but with the growing interest and progress of distributed ledgers, a new way of dealing with existing problems appeared. On the other hand, in our point of view, a lot of development groups are working in parallel on the similar topics, yet it is not clear what is going on inside. In this paper we will try to define the differences and discuss both pros and cons of using such commonly known technologies as Sovrin based upon the Hyperledger Indy technology, Civic, Jolocom, uPort and some others. Besides, we'll tackle the idea of using the SSI for inanimate object and how it can be constructed in this way.

1 Introduction

Obviously, the growing interest in blockchain technology make a lot of people think of its opportunities in variety of areas. One which is the digital identity. During last thirty years the idea of digital identity was defined and presented under different names in the following works [7,11,15]. But in the last ten years it was clarified in the name of self-sovereign identity. For clear understanding lets talk about so-called the self-sovereign identity ecosystem.

Figure 1 shows the most notable players on the field of SSI. All of these projects are focused on the problem, that today people, users of different devices, reveal a lot of private information to the third parties. That means giving the part of personal data which is not needed for some particular purpose. Usually a person does not control what is happening with his data. Third party could reuse it for own profit, manipulate it, etc. Another case is when the whole database of some entity, the company which collects and stores user accounts data on its servers, will be stolen (for example), thus leading to financial or any other personal data related issues for the customers.

To prevent the first problem the concept of self-sovereign identity appeared. As the name tells us, every person must have his or her sovereign data about him or her self, and only he or she has the right to create, use and, what is most important, control it. Besides, as we are talking about the internet communications, in many cases, especially financial and official, you need to be sure that the person you are contacting with is really a person and he or she has, for example,

S. Misra et al. (Eds.): ICCSA 2019, LNCS 11620, pp. 472–484, 2019.
https://doi.org/10.1007/978-3-030-24296-1_37

a right to pay and has enough money for that operation. In the papers authors properly described the concept and the proofs of the cryptography which is used inside.

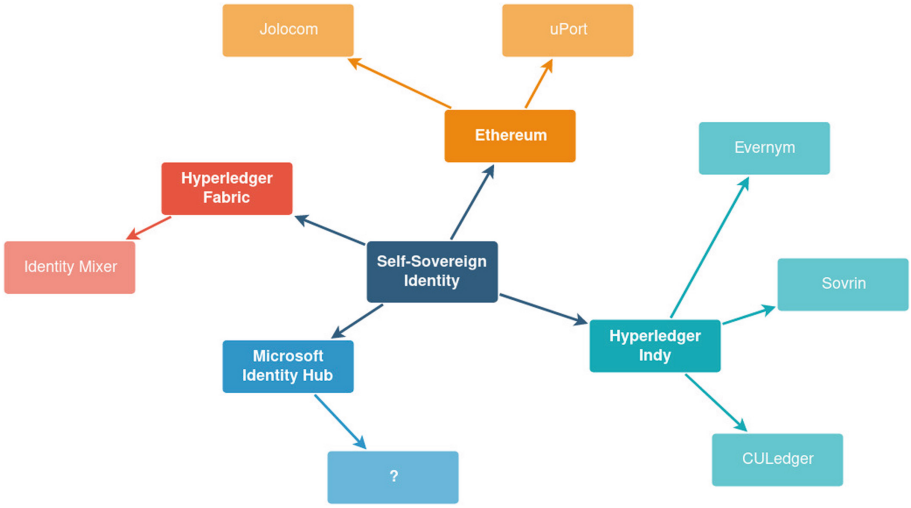

Fig. 1. Overview of the main SSI applications

Practically everything was perfect except two points:

- centralized authority;
- hash tables.

Looking at the centralized authorities which are used in the scheme, it is clear that they play crucial role in decision making. And no one can influence that, or, on the contrary, the authority can be compromised. In both cases it is the key point of the whole scheme.

The concept of distributed ledger is based on the decentralized root of trust which can not be compromised and on which everyone can rely on. Obviously, with the appearance of distributed ledgers came the idea to pass the responsibility of the centralized root of trust to them.

The second problem is the use of hash tables to store and control the key pairs (public keys, private keys) of users. The point is that they are not safe enough, though they have a lot of advantages. Developers were working on the solution for that, and as a result the W3C Community Group is now working on a specification of Decentralized Identifiers (DIDs).

From this specification we can learn that DIDs are a new type of identifiers for verifiable, "self-sovereign" digital identity [2].

2 Overview of SSI Infrastructures

First of all it should be said that SSI concept wasn't based on a blockchain at the beginning, and it was more about a certain protocol, which defines rules of interactions between independent identity agents representing an end user with its identifier. But, as it was already said before, there was a trust issue, in particular there was no way to know whether the counteragent was compromised or not. In example, if someone would be able to substitute government public keys in a storage of an agent, he would be able to make claims on behalf of government identity. So, at that point, a trust store problem was defined.

A fresh view on the matter was given with a popularization of a distributed ledger technology and blockchain in particular, since it was enabling unalterable and secure storage with a state determinated by participants based on a certain consensus rules. Thus, public key placed to the blockchain once can not be substituted, unless 51% attack was performed successfully, and every participant is able to have a full copy of a systems state.

Now, as we know main goals prospectors were trying to achieve, we can take a look on an architecture of self sovereign identity solutions and components in details.

Nevertheless there are plenty of various implementations available, most of them are having almost the same idea in mind, which is shown on Fig. 2

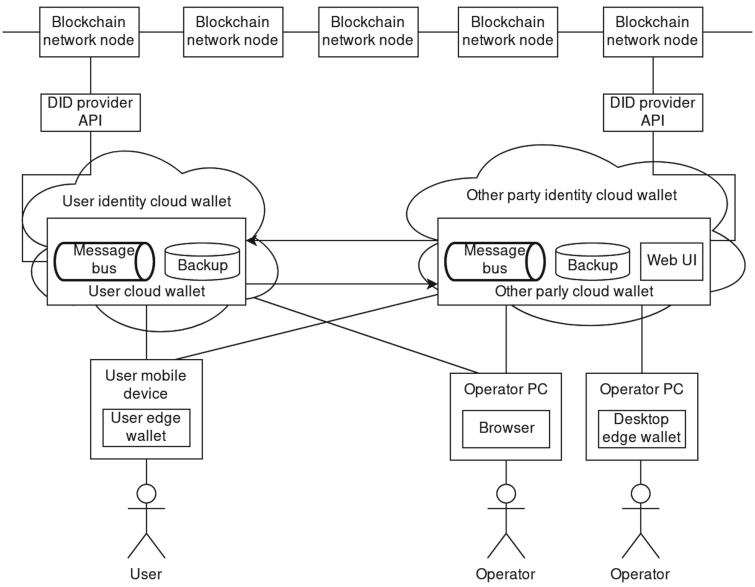

Fig. 2. Generalized scheme of interaction within SSI

Of course, blockchain comes as a basis of solution. Its main purpose is to provide a storage for end user keys and identifiers, or, in other words, to provide a public key infrastructure.

It is worth noting that a particular technology to use should be chosen with very careful consideration. Below you can find few examples (some of them are quite obvious) describing which difficulties can be met during the implementation.

Popular public blockchains, such as Ethereum [3] or Bitcoin [4], could be seen as a good candidates at a first glance. However, there's a simple, yet fundamental issue with them. There is a transaction cost for the creation of DID record, and even if it not quite big, it can bring some hassle to the end user, as he will need to find a way to buy and spend the currency. Also, transaction processing time is a lottery, especially in Ethereum, where you have to gamble on gas amount that should be put as a payment for transaction processing.

Another case could be taken from IOTA [5] snapshot mechanism. Its tangle looks quite interesting, as it allows to perform transactions on a public network really fast and free of charge. But snapshots are allowing full nodes to remove a transaction history that comes before snapshot transaction. As full nodes are storing actual tangle transaction, a public key registry would be simply eliminated.

There are some permissive networks trying to solve particular business case and eliminate conventionalities, such as mining and transaction fees, which are stimulating public participants to stay in the network in exchange to some profit. But, again, a problem of trust appears here, and you can trust the storage only if you trust the organization responsible for it. Indy [6], a particular solution from Hyperledger foundation is aiming exactly to solve this problem with DIDs. It is a public permissive network run by foundation members. In short, "public permissive" means that everyone could read from the ledger, but only ones defined as trusted members can write into it. And, actually Indy has two ledgers. First is responsible to listing all nodes, their keys and addresses, another one handles network members and their roles.

DID provider service comes on top. It is responsible for the definition of interaction rules with a blockchain. In other words, it should define a format for the data that is going to be stored in transaction block. Also DID provider defines an API for routine operations like DID creation, lookup, etc.

Next it comes to the wallet, that is actually a most important part of the system. Here is a list of basic functions it should be able to perform:

- Wallet should store user data and key pairs securely, so that no one else would be able to access them.
- Wallet should be able to find, access and store public keys of another users by their DIDs.
- Wallet should provide a way to encrypt, decrypt, sign and check signature.
- Wallet should be able to receive messages from other wallets.

The last point should be taken with attention. An ability to establish the connection between wallets is important, since pairwise DIDs are generated exactly on connection, so that connection has unique identifiers for parties. However, in

most cases user wallet is located on mobile device, like smartphone, that is not able to be permanently online or even have a public endpoint by the nature of cellular networks. So that is where cloud wallet comes into play. It can actually duplicate whole functionality of edge wallet and act like a remote walled with Web UI, but it is not really required to. The main goal of cloud wallet is to provide a public endpoint for the user and accumulate messages until they will be delivered to the edge wallet. So it can just act like an inbox. Also it can be a common case when cloud wallet handles encrypted edge wallet backups.

2.1 Authentication Protocol with DIDs

In respect of last point of basic wallet functions list, we are proposing a protocol for handshake between arbitrary client and server, where are both using wallets/DIDs to identify themselves on a web. Main goal here is to translate an identifier to an actual encrypted key to use it in symmetric traffic encryption process, like TLS/HTTPS protocol does. Suggested steps of such handshake described in the following sequence diagram (Fig. 3). The *Wallets* here could be replaced with generalized Wallet API, then both Client and Server can use it locally or remotely without the need to create an actual transaction to verify opposite side DID on their's hardware. *DLT* stands for Distributed Ledger Technology, which can be any modern ledger system, like Ethereum.

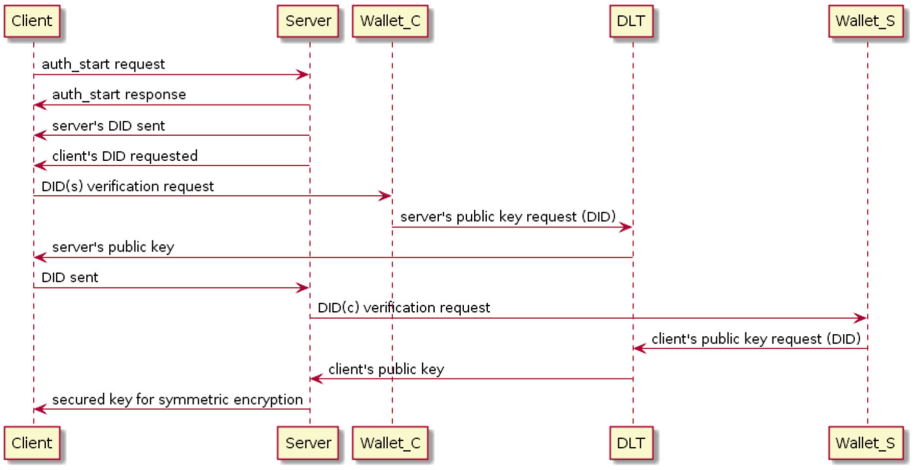

Fig. 3. Authentication protocol with symmetric encryption by DIDs

3 Anonymization Functionality

In the Privacy-ABC concept [13,18,20] every user can generate a secret key. However, in comparison with the traditional scheme of public-private key pairs

in authentication process, in Privacy-ABC there can be as many public keys for one secret as user wants. These public keys are called pseudonyms. The two very important privacy-related features are based on them. First is untraceability which guarantees that the presentation of credentials can not be linked to their issuance. In other words, that means that given two different pseudonyms, one cannot tell whether they were generated from the same or from different secret keys. Another main feature is unlinkability, which guarantees that the Verifier can not link different presentations of a given user. By generating a different pseudonym for every verifier, users can thus be known under different unlinkable pseudonyms to different sites, yet use the same secret key to authenticate to all of them. In some literature they are called Issuer-unlinkability and Verifier-unlinkability [1].

From the above the following properties of the Privacy-ABC concept can be concluded:

- Pseudonyms;
- Key binding;
- Selective disclosure;
- Predicates over attributes;
- Inspectability.

Today there is two technologies implementing Privacy-ABC concept: Identity Mixer by IBM [7] and U-Prove by Microsoft [8]. In the Table 1 comparison of main features of Identity Mixer and U-Prove is made.

Table 1. Comparison of Identity Mixer and U-Prove main features

Main aspects	Technology	
	Idemix	U-Prove
Signature scheme	Camenisch-Lysyanskaya's signature	Brand's signature
Implementation instantiation	Elliptic curves	Standard subgroup
Untracebility and unlinkability	Has both	Untraceable, but linkable between presentations
Revocation, Inspectation	Shared	Shared

In [9] a comparative analysis of the performance of these two technologies was given. From this research we can see that U-Prove is more efficient for Users operations and when the number of attributes in the credentials is big enough. But Identity Mixer has better efficiency in the rest of the cases, also with advanced presentational features.

4 Decentralized Identity Usages in Application Development

When it comes to the actual development and SSI integration into the application, two common problems of authorization and authentication are usually need to be solved, just because from the perspective of developer the concept itself is pushing for it.

On Fig. 4 you can see how a decentralized identity solution can be integrated and used in application with the classic architecture.

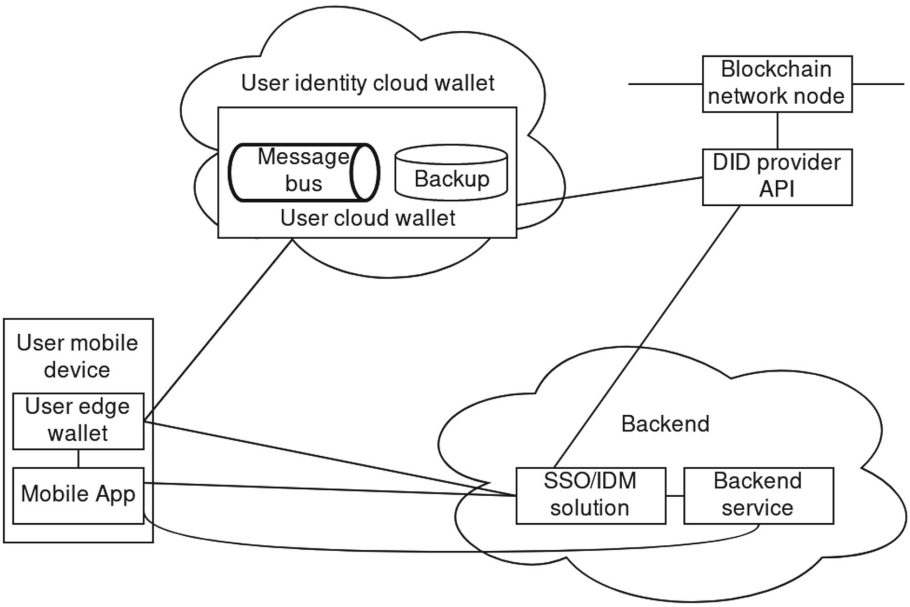

Fig. 4. Integration of DIDs on classic architecture

Taking a closer look on the components, the system has a centralized identity management solution, while the user manipulates with DIDs. Thus, there should be some kind of SSO server that will be able to communicate with SSI infrastructure and transfer claims to IDM [12]. It can be a part of identity management solution, or it can be a separate component.

This approach is similar in some ways with OIDC server intercommunication, where an identity can be shared from one resource to another, and used by target resource server to identify user. You've probably experienced that when you've been using your social account (OAuth), like Facebook or Google one, to login to another website, and it have been asking you to share some data with request originator. Moreover, an implementation of SSI data transfer to IDM can be based on OIDC protocol, so there would be a SSO server responsible to handle logins with SSI.

The main difference of centralized IDM based on OIDC compared to some SSI solution (e.g. Sovrin [14]) is that basically your mobile wallet acts as your very own identity provider, and you are choosing exactly which data you want to share with target resource, if any at all.

At final, user would be able to log into the system with the usage of SSI, and use local identity from centralized IDM to interact with resource. This approach is not completely right from the perspective of decentralized systems, but is capable to easily enable DIDs for variety of infrastructure owners using classic identity management solutions, like corporates, and also shows up a main principle of identity control.

Figure 5 represents a so-called dApp.

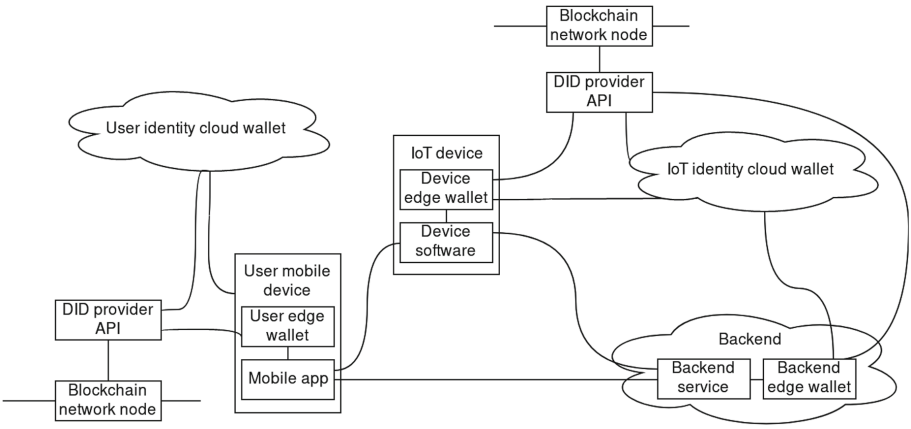

Fig. 5. The general scheme of decentralized application

So far we've been describing authentication and authorization, but another interesting area where SSI/DID solution can be used is a cryptography. These appliances are quite simple and come from basic SSI principles.

Blockchain already contains public key, so we can use it as a basis for public key infrastructure. If we know a DID of machine from the pool where we want to have mutual relationships between nodes, we are able to obtain its public key and check if machine will respond correctly to the message encrypted with that key. Also this is eliminating key exchange phase at connection initialization. Here is where an idea of device identity appears.

Another case aims at a way more global problem with certification authorities. Basically, every operating system has a package of trusted root certificates preinstalled, or such package available for it, and you may not even know or think about it. This certificate set enables a secure communications over the global network. Root authorities should store their certificates securely, issue child certificates and include them to the chain. There is no actual way to know whether authority was compromised or not, you just have to trust it (Explain here how).

5 Digital Identity in Logistic Chains

All we speak about above refers mostly to the human identity. The W3C community asserts that entities means humans or organizations. But will it work with IoT devices? At first glance, the answer is yes. In this paper we are trying to see if it is really suitable for inanimate object and how it will be working in this case (Fig. 6).

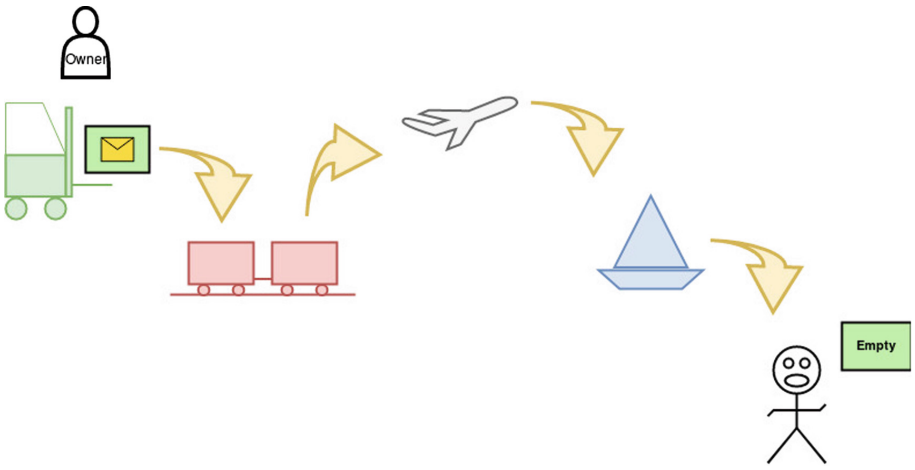

Fig. 6. Logistic chain of some item

Lets suppose we have some kind of "smart box". It can be a huge container traveling on the ship, or just a small post parcel. Each smart box has its identifier, which holds the following static information:

- the identity number;
- the weight;
- the size;
- the owner;
- properties.

At the same time as the box is smart it has some sensors and can dynamically give the information about:

- GPS coordinates;
- temperature;
- humidity;
- shaking;
- etc.

All described above opportunities of the smart box will help the owner to control cargo at each step of the chain. Throughout its journey the box will collect all the information about itself and the goods inside. Besides, if some number of smart boxes are going to join together, they can first make a decision about compatibility of there goods, and only after this check if they will group or not, if impossible.

6 Existing Solutions and Differences

There are a lot of existing SSI solutions, most of which, but not all, are based on distributed technologies and blockchain. Since there are so many of them it might take another paper to cover them so we would only focus on the most popular/famous ones. See Table 2 for some basic comparison.

Table 2. Comparison of existing SSI solutions

Solution	Aspect	
	Protocol	Source
Sovrin	Hyperledger Indy	Open-source
Civic	Ethereum	Closed-source
uPort	Ethereum	Open-source
Jolocom	Ethereum	Open-source
Veres one	Custom	Open-source
Ontology	Custom	Open-source
Remme	Hyperledger Sawtooth	Open-source

As you can see there are a lot of different approaches to the SSI using different technologies. This list is by far not complete. You can see many more solutions on the Decentralized Identity Foundation (DIF) site [28]. All of them are different of course but there are some similarities also. Rather than going through each of the solutions in detail lets just briefly go through them.

The most notable players on the market as of now are Sovrin [13], uPort [22] and Civic [23].

6.1 Source

While Civic might be the most mature project out of all existing ones - it is unfortunately closed-source. In our opinion there's no point of going through closed-source solutions as it is not completely clear what may happen inside. So even though Civic might be the best approach - we will not know until the sources are made public.

As you can see most of the other solutions are actually open-source.

6.2 Protocol

Unfortunately, as already stated above - Ethereum-based SSI solutions have a problem with every transaction being payed for. Which may not be ideal for a lot of scenarios, especially if it is used as some kind of government ID. This way either users will have to pay to use it, or the government will have to sponsor this.

Because of this a lot of consideration should be made before choosing either uPort [22] or Jolocom [24].

Same should be said regarding any of the custom protocols also - as one should investigate them in-depth before relying on them for the future use. For example this is about Veres one [25] and the Ontology [26].

There are also some Bitcoin-based SSI solutions, but clearly, given the Bitcoin architecture and general slowdown with time - using them doesn't seem like the best approach.

6.3 Standards

Another very important topic to consider when choosing SSI solution is standard compliance. The more complaint it is - the less there is a chance to be deprecated. Of course, first of all the solution should use the standard - DID for communications. And DIDs should be complaint with the DID specification [2]. Second - they should be present in the DID method registry [29]. Unfortunately Remme [27] is not present there so it is not clear if they use DIDs at all. Third - ideally public network DIDs should be resolvable by the Universal Resolver [30]. This can serve as an easy check if the DID is actually valid and complaint.

6.4 Our View

From our perspective at this point the best approach is taken by the Sovrin - this blockchain is specifically designed to solve the SSI use case, so it does not suffer from a lot of potential issues with existing blockchains. It is completely open source, with all the documentation available for developers. It is also W3C standard complaint and even works with Universal Resolver. The main downside of the Sovrin is that it is not yet mature enough to be used by everyone though.

7 Conclusion

In this article we have tried to consider the main key points of the SSI technology. Despite the fact that this technology itself has been developing for several decades, with the advent of new concepts, such as distributed ledgers, it has received both further development and new difficulties in application. We have tried to describe the most important problems that modern developers of DIDs and IoT have to face. In the future, we plan to expand the use of the opportunities described here to other cases.

References

1. Camenisch, J., Dubovitskaya, M., Lehmann, A., Neven, G., Paquin, C., Preiss, F.-S.: Concepts and languages for privacy-preserving attribute-based authentication. In: Fischer-Hübner, S., de Leeuw, E., Mitchell, C. (eds.) IDMAN 2013. IAICT, vol. 396, pp. 34–52. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37282-7_4
2. https://w3c-ccg.github.io/did-spec/
3. https://www.ethereum.org/
4. https://bitcoin.org/en/
5. https://www.iota.org/
6. https://www.hyperledger.org/projects/hyperledger-indy
7. Security Team, Computer Science Dept., IBM Research, Specification of the Identity Mixer Cryptographic Library, p. 49 (2010)
8. https://www.microsoft.com/en-us/research/project/u-prove/
9. Veseli, F., Serna, J.: Evaluation of privacy-ABC technologies - a study on the computational efficiency. In: Habib, S.M.M., Vassileva, J., Mauw, S., Mühlhäuser, M. (eds.) IFIPTM 2016. IAICT, vol. 473, pp. 63–78. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41354-9_5
10. Sanchez, J.L.C., Bernabe, J.B., Skarmeta, A.F.: Integration of Anonymous Credential Systems in IoT Constrained Environments, p. 11 (2018)
11. Wagner, K., Némethi, B., Renieris, E., Lang, P., Brunet, E., Holst, E.: Self-sovereign Identity. A position paper on blockchain enabled identity and the road ahead, p. 57 (2018)
12. https://github.com/WebOfTrustInfo/rwot4-paris/blob/master/topics-and-advance-readings/dkms-decentralized-key-mgmt-system.md
13. SovrinTM: A Protocol and Token for Self-Sovereign Identity and Decentralized Trust, A White Paper from the Sovrin Foundation, p. 42 (2018). https://sovrin.org/wp-content/uploads/2018/03/Sovrin-Protocol-and-Token-White-Paper.pdf
14. The Sovrin Trust Framework Working Group, Sovrin Provisional Trust Framework, p. 30 (2017). https://sovrin.org/wp-content/uploads/2018/03/Sovrin-Provisional-Trust-Framework-2017-06-28.pdf
15. Brickell, E., Camenisch, J., Chen, L.: Direct Anonymous Attestation, p. 24 (2004)
16. Camenisch, J., Drijvers, M., Lehmann, A.: Anonymous attestation using the strong Diffie Hellman assumption revisited. In: Franz, M., Papadimitratos, P. (eds.) Trust 2016. LNCS, vol. 9824, pp. 1–20. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45572-3_1
17. Jhttps://ssimeetup.org/
18. Garman, C., Green, M., Miers I.: Decentralized Anonymous Credentials, p. 21 (2013)
19. Fei, C., Lohkamp, J., Rusu, E., Szawan, K., Wagner, K., Wittenberg, N.: Self-Sovereign and Decentralised Identity By Design, Jolocom Technical White Paper, p. 10 (2018)
20. Rannenberg, K., Camenisch, J., Sabouri, A.: Attribute-Based Credentials for Trust, p. 395. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-14439-9
21. Neven, G.: A Quick Introduction to Anonymous Credentials, p. 4 (2008)
22. https://www.uport.me
23. https://www.civic.com/
24. https://jolocom.io/
25. https://veres.one

26. https://ont.io
27. https://remme.io/
28. https://identity.foundation/
29. https://w3c-ccg.github.io/did-method-registry/
30. https://uniresolver.io/