Journal of Modern Technology and Engineering
Vol.5, No.3, 2020, pp.189-211

SELF SOVEREIGN IDENTITY BASED MUTUAL GUARDIANSHIP

Yusuf Dündar[1*] , **Isa Sertkaya**[2]

[1]National PKI Lab TÜBİTAK BİLGEM UEKAE, Kocaeli, Turkey
[2]MCS Labs & BCLabs, TÜBİTAK BİLGEM UEKAE, Kocaeli, Turkey

Abstract. While digital identities are increasing day by day, physical identities have started to be digitalized. With these two worlds getting more intertwined than before, new identity management schemes become crucial. Self Sovereign Identity (SSI) is a new approach that develops promising solutions on the longstanding identity problem. SSI schemes come into play in this regard and propose a new model with certain principles such as privacy, Interoperability, consent. For the mass adaption of such new solutions, solutions already used in daily life should also be considered. In this context, indirect SSI controls, such as guardianship, are essential for SSI's mass adoption. In this study, we focused on indirect SSI control, particularly custodianship. Custodianship handles a dependent's (such as an underage or immigrant child) identity management by her guardians. The proof of concept custody framework is designed based on Sovrin Foundation's SSI solution. This is fundamental since, in joint custody, management of the dependent's identity must be jointly carried out by two guardians. Modeling of the framework was carried out on a sample SSI network, and performance analysis was performed with existing systems. These analyses show that the proposed framework can be integrated Sovrin SSI framework without overloading the existing systems.

Keywords: self sovereign identity, indirect SSI control, privacy, guardianship, custodianship, blockchain.
AMS Subject Classification: 68M14, 68M25, 94A60, 68P27.
Corresponding author: Yusuf, Dündar, TÜBİTAK BİLGEM UEKAE, Kocaeli, Turkey, Tel.: +905377111777, e-mail: *yusuf.dundar@tubitak.gov.tr*
Received: 18 August 2020; Accepted: 23 October 2020; Published: 24 December 2020.

1 Introduction

Identity is defined as "*The set of characteristics that make you who you are.*" in the Rountree (2012). To understand the identity concept in detail, one needs further clarification of the realm and context where the identity is used.
Real identity enables the holder to have access to financial, health and/or public services. In fact, as clearly defined in Article 6 of the Universal Declaration of Human Rights, everyone has the right to recognition everywhere as a person before the law (UN, 1948). Unfortunately, it is estimated that over one billion people on earth can not prove their identity according to World Bank (2018).

Digital identity notions evolved with the rise of the Internet. They have gone through four main stages to evolve today's systems. Four stages of digital identity are defined by Christopher Allen in Allen (2016). At the beginning of the Internet, only the centralized authorities have a right to publish digital identity. In the later years, Certificate Authorities (CAs) show up and create a hierarchy between the centralized authorities. However, there are still core authorities in the root that holds up the power. The need for interoperability between the different website identities caused Federated Identity to appear. Federated Identity relies on Identity Providers (IdP). IdPs are the system component that creates and manage the identity information. The

user is first required to register with an IdP. Then the user can authenticate an application with these IdP-owned identities. SAML (Maler et al., 2005) and LAP (Liberty Alliance, 2005) are the open-source projects that work on the federated identity model. However, it is still a centralized system and administrated by federated commercial companies. Thereby the user-centric identity notion was born. User-centric methodologies tend to work on two elements: user consent and interoperability (Allen, 2016). Several initiatives like OpenID Connect (Sakimura et al., 2014) provide more simplicity for individuals by providing Single Sign-On (SSO) functionality. OAuth (Hardt et al., 2012) is an open-source standard that allows secure authorization.

If we look at how online identity currently works, we see every person has many identities set at different companies and organizations. One person has multiple identities set depending on the social context like work, hobbies, etc. Each of these administrative identity systems owned by an organization that provides it, and we use usernames and passwords to authenticate ourselves. Most websites use centric approaches like "Sign-in with Google" or any other big companies. This will lock us up in their ecosystem. The consequences are that the users are not in control, data breaches and identity fraud are prevailing issues. News about identity data breaches has been increasing in recent years. %27 of 1107 people surveyed faced data breaches in 2018, according to Security.org (2018). Prominent examples of big identity thefts involves big companies such as Yahoo (Spangler, 2017), Dropbox (Gibbs, 2016) and LinkedIn (Hackett, 2016). Another concern about centric approaches is that if the trusted authorities are compromised in some way, those identity data can be used in negative ways. To give an example of this Single-Point of Failure situation, in 2011, a Dutch Certificate Authority was hacked and allowed supposedly secure encrypted data going across the Internet to be intercepted and accessed by hackers (Adkins, 2011). These lead to the need for a new base layer framework for identity.

Identity management is still an evolving and open issue, even if technological developments are rapidly changing.

All these different solutions evolved and came into the present day. They all have one purpose, to solve the ongoing identity problems on the Internet. These problems can be summarized as follow with reference to Windley (2017).

- **The Proximity Problem:** The problem is not known who the people we communicate over the Internet are in real life.
- **The Scale Problem:** As our digital assets increase in the online environment, it becomes more challenging to manage.
- **The Flexibility Problem:** Problems arising from the fact that existing internet identity solutions cannot work with each other
- **The Privacy Problem:** Collecting and sharing digital identity data in specific organizations such as Service Providers creates privacy problems related to our identity. These digital data have become an open target for cyber attacks. Simultaneously, the provision of digital identity services through individual institutions constitutes a Single Point of Failure.
- **The Consent Problem:** Under no circumstances should digital identity data be shared without the consent of the person.

In recent years, with the rise of blockchain technology, there are new opportunities for a fully decentralized identity system. Using the distributed ledger technology (DLT) and decentralized identity management systems, finally, we enable any entity to create and manage their own identifiers on any number of distributed, independent roots of trust without introducing a centralized authority or a single point of failure (Reed et al., 2020).

Therefore, the self sovereign identity notion was born. Self Sovereign Identity (SSI) is a step forward from User-Centric Identity and could be the solution to the identity problems in the centralized identity system. SSI is defined as

> "The digital movement that recognizes an individual should own and control their identity without the intervening administrative authorities."

in Sovrin (2018), one of the leading non-profit organization in SSI.

Nevertheless, SSI is described in many different ways; it has certain principles to follow. These guiding principles provide a better perceive the definition of SSI. Christopher Allen stated these ten key principles of self sovereign identity as follows (Allen, 2016).

- **Existence:** In the heart of SSI there must be an independent individual exist.
- **Control:** The person must have a full control of his/her identity.
- **Access:** The person must have a continuous access of his/her data.
- **Transparency:** Underlying algorithms that are used in the system should be free and open-source.
- **Persistence:** The data should remain for the time the person wants.
- **Portability:** The system should not restricted with singular third-party. The data should be transportable.
- **Interoperability:** Data should be available as widely as possible.
- **Consent:** The users must have consent for the use of their data.
- **Minimalization:** The users should not need to overexpose their identity.
- **Protection:** The system must always protect user identity.

SSI solutions should realize the principles given above. These principles are crucial in order to overcome the shortcomings features of currently widely used identity management systems. With the help of these principles, many different working groups, foundations, or companies are trying to make SSI real-life implementation. uPort (Lundkvist et al., 2017), BlockStack (Ali et al., 2016) and Sovrin (Reed et al., 2016) are the leading developers to implement SSI solutions. Although there are multiple SSI implementations, they all use the same infrastructure on account of Interoperability.

One of the important aspects of this infrastructure is the "Verifiable Credentials (VC)". All the SSI solutions utilize World Wide Web Consortium (W3C) verifiable credentials data model (Sporny et al., 2019). W3C verifiable credentials working group defines verifiable credentials as

> "It is a tamper-evident credential that has authorship that can be cryptographically verified."

in Sporny et al. (2019). VCs are the center of the SSI systems. They are managed by the user him/herself. In physical life, we also have credentials such as college degrees, passport, etc. These all help verify that we have some attributes. The digital credential is equivalent to these credentials in the online environment. VC might consist of information related to the subject, credential type, issuing authority, and some constraints the same as physical credentials. However, apart from these pieces of information, VC have additional attribute such as digital signatures (Sporny et al., 2019). Thereby this makes VCs cryptographically secure, privacy-enhanced, and more trustworthy.

Another aspect of the infrastructure is the blockchain. Blockchain technology has become more prevalent in recent years with cryptocurrencies. Bitcoin was introduced with the paper *"Bitcoin: A Peer-To-Peer Electronic Cash System"* by Satashi Nakamota in 2008 (Nakamoto, 2008). Since that day, many digital payment applications and currencies were developed. Later, with smart contracts, the way for different finance applications to use blockchain was opened. But in recent years, with the rise of the distributed databases, blockchain was used beyond currency and finance areas. It becomes multidisciplinary technology and could be uses in different areas such as art, government, big data, digital identity verification, and distributed apps.

Last but not least, SSI relies on distributed ledger technologies. DLT defines as follows

> "DLT refers to a novel and fast-evolving approach to recording and sharing data across multiple data stores (or ledgers)."

in Natarajan et al. (2017). This technology allows for transactions and data to be recorded, shared, and synchronized across a distributed network of different network participants.

All these Infrastructure aspects constitute that Self Sovereign Identity application could be adaptable in real life. To that end, Hyperledger Indy was launched as an open-source project under the Linux Foundation in 2018 (Hyperledger, 2018). The purpose of this project provides open-source tools and libraries for people and institutions to develop inter-operable SSI applications. Later Hyperledger Indy project extended and spin-off new projects such as Hyperledger Ursa (Linux Foundation, 2018) and Hyperledger Aries (George, 2019a). Sovrin Foundation, established under the company Evernym provides real-life running application and use cases using their Sovrin SSI Framework and Hyperledger technologies (Sovrin, 2018). The first version of the Sovrin Framework was officially released on June 28, 2017, under the name *Sovrin Provisional Trust Framework* (SPTF), and the second version was released on March 27, 2019, under the name *Sovrin Governance Framework* (SGF) (Sovrin, 2019b). SGF is developed by *Sovrin Governance Framework Working Group* (SGFWG). But it is open to everyone's contribution. The British Columbia government announced OrgBook BC project in 2019 that using Sovrin Network (British Columbia, 2019). This project aims to modernize services for the citizens as part of the Digital Government Strategy. On the other hand, the studies to standardize the technology notions used in SSI, such as Verifiable Credentials (Sporny et al., 2019) and Decentralized Identifiers (Reed et al., 2020) continue with the several working groups.

Even if SSI has promising features such as user's ownership of his/her identity, privacy enabled authentication, selective disclosure; these features bring new responsibilities to the identity owners. User credentials bind to cryptographic keys and secrets. Thus, in order to protect privacy concerns, users need to manage these credentials properly. This leads to the usage of digital wallet application and agents.

If the SSI is desired to be used worldwide, it must handle all real-world use cases. But this idea reveals the problem, what will happen to those who cannot control their own data? For instance, older people or underage children who cannot use digital wallets to manage their credentials. To solve this problem, we have to examine the identity control mechanism in SSI systems.

In Sovrin Network, there are defined three different indirect identity control system for solving the problem mentioned above (Sovrin Guardianship Task Force, 2019). These can be shortly defined as follows and will be further studied in Section 3.

1. **Delegation:** It is the case that an Identity Owner that acts on behalf of another Identity.

2. **Guardianship:** An Identity Owner who administers Identity Data on behalf of a Dependent

3. **Controller:** An Identity Owner that is responsible for control of another Entity

These relationship models help manage credentials on different use cases. Thus it is an essential part of the Sovrin Framework.

In this study, we focused on the guardianship model and explored it would be possible to use Sovrin Network, Hyperledger Indy, for the joint custodianship use case.

Related Work. In this section, we are going to summarize self sovereign identity proposals and studies within identity guardianship. We will list master's and doctoral thesis related to our subject. Then we will mention other works, white papers, and website articles.

In Dunphy et al. (2018), Paul Dunphy from Cambridge describes the challenges for Distributed Ledger systems. In Davie et al. (2017), Uwe Der explores SSI systems' opportunities and challenges briefly. With the rise of distributed technology, many SSI solutions came out. In Abraham (2017), van Wingerde (2017) and van Bokkem et al. (2019), SSI solutions such as Sovrin, Blockstack, Multichain, and uPort compared in terms of blockchain types, key management system, and Proof-Of-Work system. uPort (Lundkvist et al., 2017) is an Ethereum blockchain-based, open-source SSI solution released in 2017. It is based on Smart Contracts. One of the disadvantages of uPort is that it does not allow users to revoke their consent. Similar to uPort, same year Civic Wallet came out. Their mission to give people control over their cryptocurrency data (Civic, 2017). Apart from digital currency-related solutions, there are consortium-based solutions about identity management such as id2020 (id2020.org, 2018). Their mission to give an identity who does not have an identity in the world using blockchain technologies. Also, there are several works about identity. In Ruff (2020), Timothy Ruff propose a Self-Sovereign Student ID. Smart Custody proposed by Christopher Allen to protect the personal digital assets using cold storage (Allen & Appelcline, 2019).

Sovrin is a foundation based SSI solution using permissioned blockchain. It was proposed in 2016 by Sovrin Foundation (Tobin & Reed, 2016). Sovrin Foundation is a non-profit organization founded by Evernym Company. They are more prominent than other projects due to being open-source and non-profit. In Windley (2016) and Reed et al. (2016), it was mentioned technical details about Sovrin. It is based on Hyperledger Indy (Hyperledger, 2018) and Hyperledger Aries (George, 2019a) technology. It also use Decentralized Identifiers (Reed et al., 2020), idemix Anonymous Credential (Camenisch & Van Herreweghen, 2002) and Verifiable Credentials (Sporny et al., 2019) as a underlying technologies.

There are more SSI studies on Sovrin Network. Such as in Kondova & Erbguth (2020), Galia Kondova studied the compatibleness of the Sovrin network with the EU GDPR law. In Weller & Dijksman (2019), Daan Weller analysed the Web of Trust and compared it to the Sovrin Network.

Recently, Delegation and Guardianship is a rising hot topic on SSI systems. Various solutions have been proposed to implement the Indirect SSI control in real life. Camenisch et al. (2017) presents the first delegatable anonymous credential system that is practical in 2017. Under the Sovrin Foundation, the Guardianship Working Group was established in December 2019. They have published the first study in Sovrin Guardianship Task Force (2019) to form the basis for further work. They studied two real world cases. Nevertheless there is a lot of other use case needs to be studied. Still, a robust final solution has not yet been revealed regarding the guardianship.

Our Contributions. Considering the studies mentioned in the previous section, we have discussed the problems of *Self Sovereign Identity based guardianship* and put forward clear definitions of the problems. In order to model the solution of these problems, joint custody, which is given as a result of divorce, was chosen as an example from real life. Through this use-case, a framework design has been made in which the identity of one person can be managed jointly by two different people. Within the framework, which was designed to overcome this use-case's difficulties, the permissions and restrictions of the guardian were clearly defined. In order for the framework to be used in an SSI system, it was put on Github in both human-readable

and machine-readable form. Using Hyperledger Indy and Hyperledger Aries libraries, custody credentials were created in Sovrin's SSI network using this framework. Credential's life cycle steps were tested sequentially, and it was proven that the created credential works properly in existing SSI systems. Finally, we analyzed the solution we implemented in detail. By comparing with existing systems, we have shown that custody credentials can be used in existing systems without any extra load.

Organization. Section 2 will give the necessary definitions about Self Sovereign Identity and its multilayered structure. Later, the working mechanism of credential management is described. Section 3 is focused on Indirect SSI controls. After we briefly explain what the Indirect SSI concept is, Guardianship, Guardianship life cycle, risks, and challenges are described. Section 4 contains our Custodianship Trust Framework. We briefly explain what custody is. After that, we formalized our use case joint custody. Lastly, we described the main construction and implementation of the Custody Framework. In Section 5, we analysed the proposed framework. Comparative analysis of our framework and existing system was made. Finally Section 6 summarizes and concludes the manuscript.

2 Sovrin: Self Sovereign Identity

SSI is the most advanced point of identity management systems. It is a truly decentralized system. Although many different companies working on SSI, Sovrin Foundation was taken as reference in this study. The study was conducted on the Sovrin Network. Therefore, the technical infrastructure of SSI will be explained over the Sovrin Network.

2.1 Sovrin Architecture

Sovrin architecture consists of 4 layers. This architecture is called the Trust over IP (ToIP) Stack. It is defined in Davie et al. (2019). ToIP is designed to fully support digital guardianship. Four layers can be seen in Figure 1. In this section, we will examine these four layers of SSI architecture as follows:

- **Layer One: DID Networks**: In this layer, which is the lowest layer, a decentralized network is provided by using public permissioned blockchain. Hyperledger Ursa crypto library (Linux Foundation, 2018) is used to perform cryptographic transactions on this blockchain. The blockchain used in this layer can be considered as a ledger. Each record on this ledger is called *Ledger Entity*. These Ledger Entities can consist of public keys, credential schemas, and credential definitions. This network also supports Decentralized Identifiers. All these records must have a globally unique DID number. It can be known which institution these DID numbers represent, such as government agencies. But this is not mandatory. It can be a pseudonym. Thus confidentiality is provided between two actors. Also, for privacy and security, DID wallets do not contain credentials. Thus, DID does not contain the personal information of any person. Layer One is a fully cryptographic distributed network infrastructure.

- **Layer Two: DIDComm**: There are digital wallets in layer two. Wallets provide secure peer-to-peer communication via agents over DIDComm protocol. It is the layer where the basic guardianship tasks are performed. Because of the management of digital wallets, and therefore, private keys happens in this layer. The Guardian must create a separate wallet to check the keys of the dependent. Thus, the distinction between his wallet and the dependent's wallet is made, and many possible problems are prevented.

- **Layer Three: Credential Exchange**: It is the layer where the human trust comes into play. The exchange of digital credentials between Issuer, holder, and verifier takes place

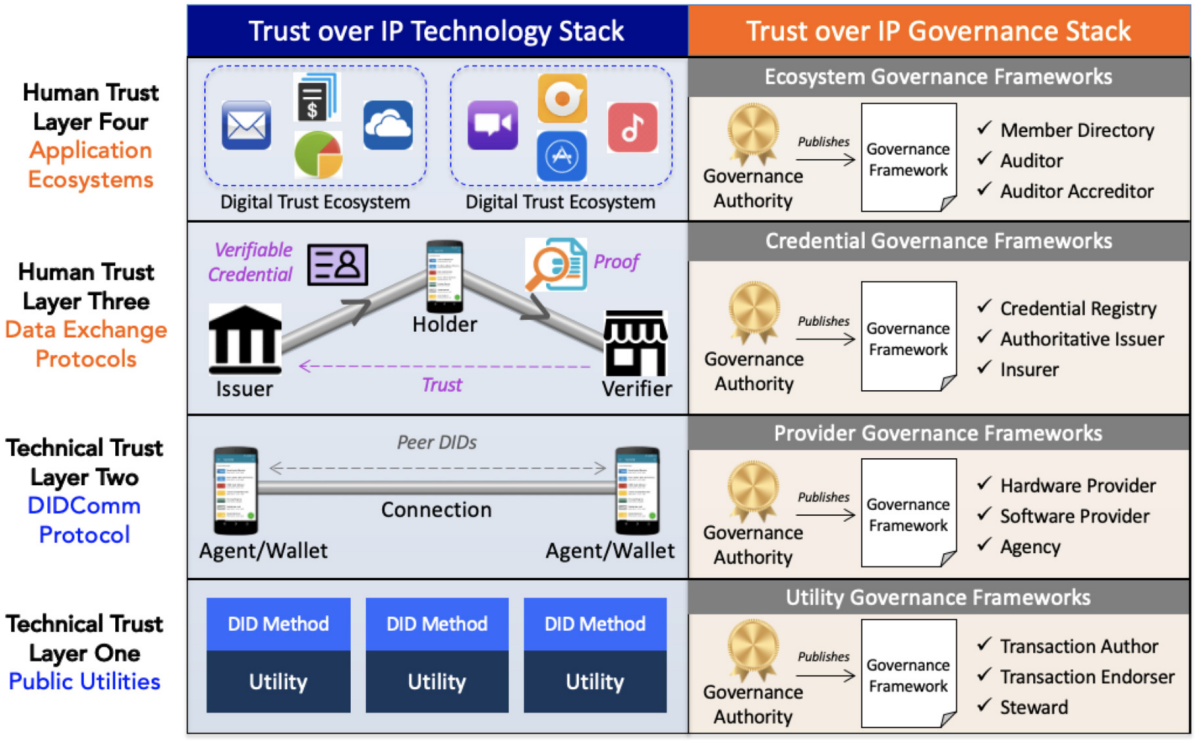

Figure 1: The four layers of the Trust over IP stack (Davie et al., 2019)

in this layer. The layer one is linked with Layer two as its exchanges are made via the DIDComm protocol. Verifier can easily verify the public key by checking from the layer one to verify Issuer's DIDs.

In this layer, Sovrin uses the Hyperledger Aries library (George, 2019a). Hyperledger Aries is an open-source library created to perform agent tasks such as providing the endpoint service to clients. Using Aries, people can develop agent applications for SSI. Aries library is a platform independent library, as seen in Figure 2. It works compatible with Sovrin or other SSI networks. The management of keys and credentials takes place through this library. Hyperledger Indy library, which enables communication with a ledger in its infrastructure, is used. The Indy library acts as a resolver for DIDs, schemas, and credential definitions.

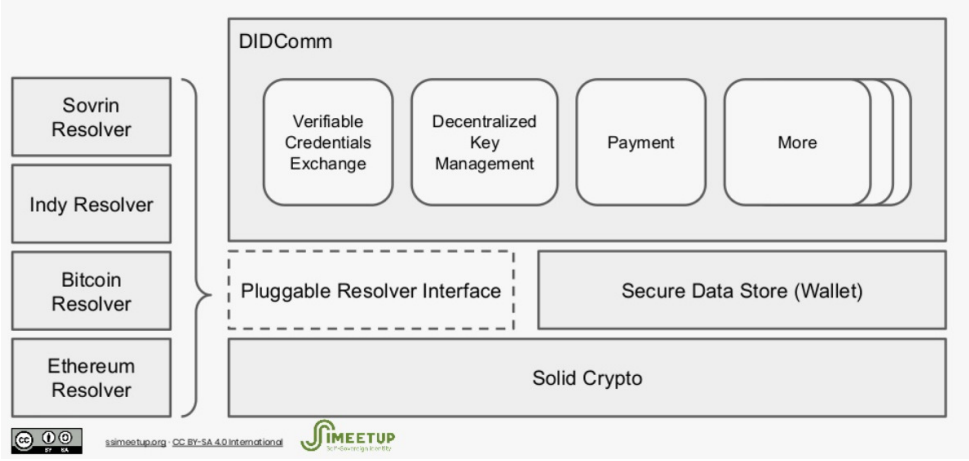

Figure 2: Hyperledger Aries Architecture (George, 2019b)

- **Layer Four: Governance Framework**: It is the layer that added human control to cover the first three layers. The only technology in this layer is the definition of verifiable credentials reserved exclusively for the use of authorized institutions such as the government. This layer at every stage of the guardianship is the source of all legal processes. Thus, it is ensured that the processes are carried out correctly.
  A governance framework structure that is special to guardianship has been created within Hyperledger Aries in 2019 (Hyperledger, 2019b). Aries agent framework has libraries created for different software languages such as Python, .NET, GO. In this study, Aries Agent Python library (Hyperledger, 2019a) was used. Aries has ZKP-capable verifiable credentials implemented using Ursa foundations. It also provides a Decentralized Key Management System (DKMS) specifications via Hyperledger Indy. Sovrin Foundation is one of its most important contributors. One of the essential tasks of the agents is verifiable information exchange. Let us take a closer look at how this workflow is in the next Section.

2.2 Sovrin SSI Credential Management

The roles and information flow in the verifiable credential ecosystem are as follows:

- An issuer issues a verifiable credential to a holder. Issuance always occurs before any other activities involving a credential.
- A holder might transfer one or more of its verifiable credentials to another holder.
- A holder presents one or more of its verifiable credentials to a verifier, optionally inside a verifiable presentation.
- A verifier verifies the authenticity of the presented verifiable presentation and verifiable credentials. This should include checking the credential status for the revocation of the verifiable credentials.
- An issuer might revoke a verifiable credential.
- A holder might delete a verifiable credential.

Note that the order of the actions above is not fixed, and some actions might be taken more than once. Such action-recurrence might be immediate or at any later point. Figure 3 demonstrates life cycle of a single verifiable credential.

Self sovereign identity relies basically on attributes. An attribute explains in Doerk & Helper (2019) as “what qualifies a person, without necessarily being unique to that person.” These are elements such as gender, weight, height, etc. Attributes can also be verifiable credentials.
Verifiable credential has three entities. These are

- Issuer (I)
- Holder (H)
- Verifier (V)

The simple workflow between these entities is explained below with reference to Lodder et al. (2019).
To simplify, the example has one credential and single I, H, and V entities. But in real life, Entities can be multiple, and the holder can have multiple credentials.

- First of all, the Issuer must create a “Credential Definition” from the schemas registered in the ledger. Credential Definition is a record containing the necessary public parameters for a credential. These parameters are as follows: Signature parameters and Revocation Registry parameters.

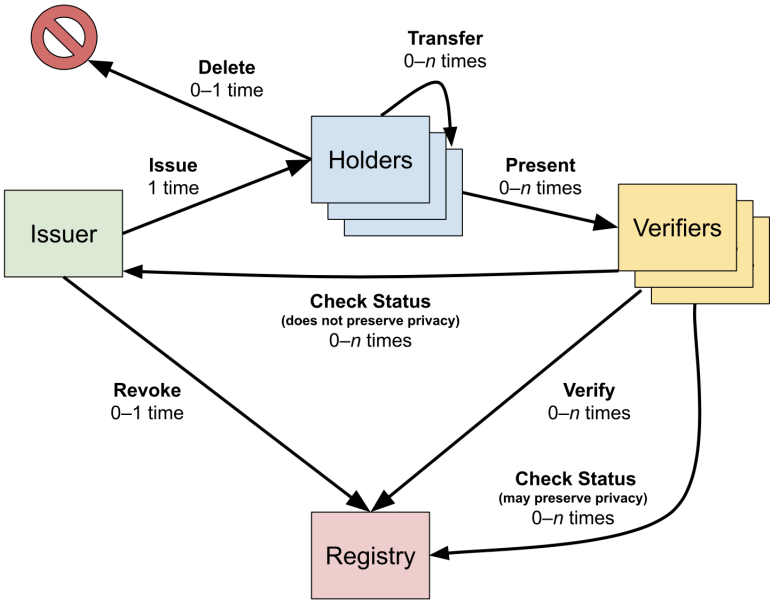

Figure 3: Life of Single Verifiable Credential (Sporny et al., 2019)

- Issuer then publishes the credential definition on the ledger.
- Holder retrieves the credential definition from the ledger.
- Holder prepares credential offers by adding some attribute like a master secret.
- After that holder sends a credential offers to Issuer to get a credential.
- Issuer receives this offer, fills the attributes written in the schema, and publishes the credential to holder.
- Holder verifies the attributes in the credential and completes the signature.
- Verifier sends a proof request to the holder that the desired attributes are in it. The proof request contains a set of requested attributes and disclosure predicates. The predicates may be equal-to, not equal-to, greater-than, less-than, or set-membership.
- Holder prepares and sends the proof to the holder.
- Verifier can verify the proof via the necessary information on the ledger.

3 Indirect SSI Control: Guardianship

SSI systems aim to manage digital and real identities using agents and hence digital wallets that control verifiable credentials. In the road of mass adaption of these SSI schemes, these agents and digital wallets must be available to everyone and be user-friendly. Furthermore, SSI schemes should necessarily allow almost all current utilities supported in current real-life identities. Among these, indirect SSI control, which can be loosely defined delegation of one's identity to be managed by a trusted one on her behalf, is probably the most crucial one.
The need to cover everyone is best explaining why there is a need for Indirect Control in SSI systems. 3.3 billion people in the world live without access to Internet (Kemp, 2019). This number includes 1.9 billion children (Sovrin Guardianship Task Force, 2019). This shows that when designing SSI systems, we need to reconsider the functioning of relations between identity holders and other processes.

Considering the person's entire life from birth to death, any scenario should be operable in these systems. For instance, managing an underage child's identity may require more than a simple delegation method. Because the child cannot give delegation over an identity that she does not know how to control.

The Guardianship subject has been an important component of SSI from the very beginning. For this purpose, the Sovrin Guardianship Working Group, established under the Sovrin Foundation, aims to provide a basis for future studies on this subject. Considering the real-life restrictions, Sovrin Foundation is looking for a solution to this problem without giving up SSI's basic principles. For this purpose, three indirect identity control models are defined and summarized in Table 1. In this section, guardianship, one of these three models, will be examined in detail.

Table 1: Primary distinctions between the three types of identity control relationships (Sovrin, 2019a)

	Delegation Relationship	**Guardianship Relationship**	**Controller Relationship**
Exists between two Identity Owners	Yes	Yes	No
Exists between an Identity Owner and a Thing	No	No	Yes
Both parties control their own Private Keys	Yes	No	No
Who authorizes the relationship	Delegator	Dependent or a legal representative of the Dependent	Thing Controller (or legal owner of the Thing)
Authorization mechanism	Delegation Credential (may be backed by legal agreement)	Legal agreement (may be backed by Guardianship Credential)	Thing Controller Credential
Who has legal responsibility	Depends on the relationship	Guardian (serving as information fiduciary)	Depends on the relationship

The concept of guardianship is in everyone's life with situations such as raising a child or caring for the elderly. Many of them are actions we take without a legal obligation. However, there are different scenarios where guardianship is needed. As an example, when people do not have access to the Internet or as a result of legal obligations. In cases where people do not have access to the Internet, people living in refugee camps can be real-life examples. Charities act as guardians for these people. Thus, it is ensured that it can get the necessary help in the system. When the refugee becomes able to manage their own digital assets, this guardian relationship is terminated. Legal obligation can be given as an example of guardianship by a minor's family or court orders. Guardianship includes a wide range of real-life scenarios. SSI has to meet all these scenarios. In these scenarios, there is not always the same level of guardian relationship. Depending on the dependent's condition, the responsibilities of the guardian can be divided into 3.

1. **Full guardianship:** It is a situation where the Dependent cannot do any action on her/his own.
2. **Protective guardianship:** Situations where the dependent person can transact with the help of the guardian person.
3. **Supportive guardianship:** It is the situation where the Dependent can operate on her/his own, but prefers to have a guardian with her/him.

Sovrin framework is designed with the principle of privacy by design. This principle also addresses the issue of privacy between Guardian and Delegate which is explained in (Sovrin, 2019c,

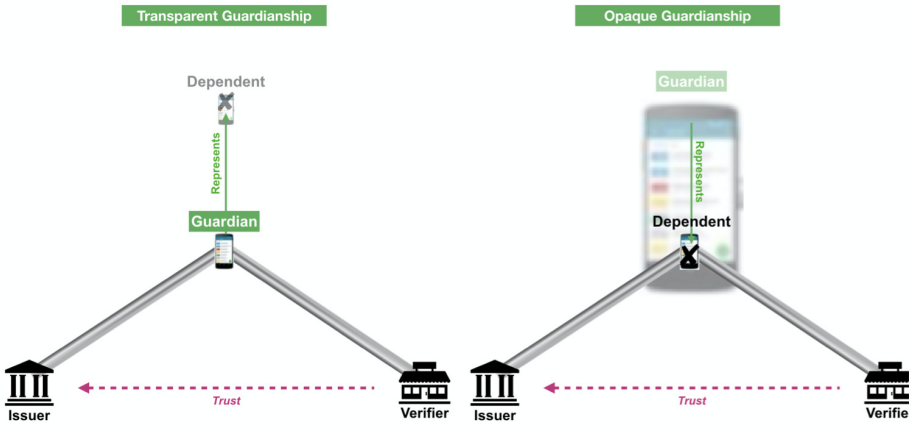

Figure 4: Transparent vs. opaque guardianship (Sovrin Guardianship Task Force, 2019)

Section 2.10.7) as follows:

> "**Guardian and Delegate Confidentiality.** The use of a Guardian or Delegate may be confidential information and shall only be disclosed with the authorization of the Identity Owner and of the Guardian and/or Delegate."

That is, the established guardian relationship does not have to be disclosed to the other party. The verifier may not be aware of this relationship. However, this situation may have to be disclosed in the guardianship established with legal obligations. The verifier may also want to verify the identity of the guardian. Considering this situation, we can say that there are two types of guardianship, as shown in Figure 4 in terms of privacy.

1. **Transparent guardianship:** In transparent guardianship, it is often known who the guardian is. Verifier often verifies the identity of the guardian. An example of this is the guardianship between a child and his family. This relationship does not need to be hidden because it is known to everyone that the child has a guardian.

2. **Opaque guardianship:** In the opaque guardianship, the verifier is not aware of a guardian's existence.. Opaque guardianship offers more privacy. An example is mental health guardianship. In this relationship, the guardian's existence is hidden because of the possibility of discrimination in the dependent's social life.

3.1 Guardianship Life Cycle

Various guardianship method can be established. So there may be different life cycles. However, it is possible to implement guardianship without affecting existing systems' operation by using schema definitions specially designed for guardianship instead of standard schema definitions, which is the solution followed in this study. So this means implementing a guardianship means issuing specific digital credentials for both the guardian and the dependent. However, these credentials define a relationship rather than describing the characteristics of individuals. The high complexity of designing the guardian relationship is due to the wide range of real-life applications. This situation includes many risks that need attention. Therefore, it is necessary to be very careful when designing. The guardianship life cycle is broadly illustrated in Figure 5 and can be summarized as follows.

1. **Inception:** The first step is to determine the need for a guardian relationship. Especially if the guardianship will be established due to a legal obligation, the steps of this will be longer. On the contrary, when an emergency guardianship is required, this relationship will

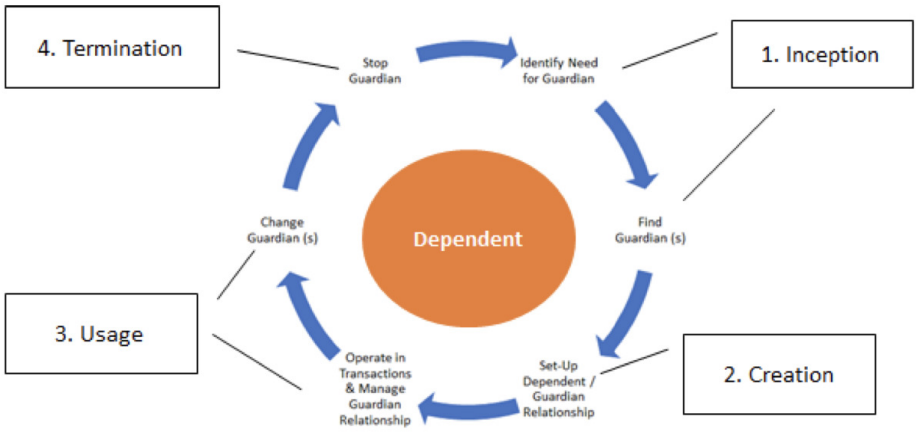

Figure 5: The four major stages in the guardianship life cycle (Sovrin Guardianship Task Force, 2019)

need to be established more quickly. This step ends when the guardianship is confirmed to be in need. Usually, this approval is done by receiving the consent of the dependent.

2. **Creation:** The second step is to create the necessary credentials for guardian and dependent. These credentials conform to the W3C verifiable credential standard. These credentials use a different schema than normal credentials. This schema is specially designed for guardianship. In most cases, the schema is prepared by frameworks in accordance with legal restrictions. But in some cases, a schema may need to be created at this step. In such a case, the schema must have defined the following items.

   - Guardian's rights and responsibilities
   - What identity data can the Guardian control
   - Limitations on the guardian's permissions.

   This phase ends when credential is created and ready for use.

3. **Usage:** This step includes real-life scenarios. The credentials created in the previous step are used by the guardian and dependent person at this stage. Credentials can be revoked if necessary, depending on the state of the relationship. This stage continues as long as the guardian relationship continues.

4. **Termination:** When Dependent is able to control his own digital assets, he has the right to end the guardianship. This is done by the revocation of the credentials given for guardianship. In this way, the authority of the guardian to access cryptographic keys is eliminated.

3.2 Risks and Challenges

In the Guardianship, Dependents are generally vulnerable people. This may lead to many risky situations. We will examine these challenges and summarize possible solutions.

- **Inherent Risks**: Inherent risks are very general and primary structure risks. Therefore, legal actions may be required to prevent them. As an example of these actions, consent approval may be required for the guardianship credential. A reliable framework that has implemented best practices should be used.

- **Violating The Trust Relationship**: In the guardianship, guardian acts in the best interest of the dependent. But this concept of best interest is a subjective. As a possible solution to this risk, a certification requirement may be introduced to become a guardian. Simultaneously, it is possible to set the operation to a standard by using a robust framework.

- **Impersonation And Commingling Of Identity Data**: Guardian can benefit by using dependent's trust. Several measures can be taken to prevent abuse of dependent's trust.

  - The Guardian definitely needs to create a separate wallet for dependent.
  - Dependent's wallet should always be auditable.
  - During risky transactions, verifier is always aware of the guardian relationship and needs to confirm the guardian's identity.

- **Complexity, Conflict, And Competition**: Guardianship can quickly become complicated because a person can have more than one guardian. This means multiple wallets and different credentials that need to be managed. At the same time, guardians may be using different frameworks.
  To prevent risks arising from such situations, frameworks should be designed to work with each other.

In a guardianship, there are challenging element. There are technical difficulties in representing the different social contexts, which are summarized below.

- **Differences in relationships**: The relationship between the guardian and dependent is not stable and changes over time. There must be a special trust relationship between these two. This trust gets stronger over time. But its degree varies from relationship to relationship. For example, for a child in a refugee camp, perhaps more than one guardian's approval is required to leave the camp, while only one guardian's approval is sufficient to get food. Each guardian can have different permissions. It should be designed with these differences in mind.

- **Differences over time**: The Guardianship degree may not always be the same. It can gradually decrease as it will increase. For example, the relationship of an older adult with dementia with the guardian will increase as the disease progresses. Perhaps there is a supportive guardianship at first; then, it will turn into full guardianship. The opposite can also be given as an example. Another example, the child in the refugee camp can grow and receives education over time, and become fully manageable of his own digital assets. Thus, he no longer needs a guardian.

- **Differences in online and offline contexts**: Many of the examples we gave about guardianship were examples from a face-to-face relationship. But it does not always have to be this way. Online guardianships can also be designed. In this design it should support the same processes as offline guardianship. Therefore, guardianship actions should use a common framework, whether offline or online. Thus it becomes more manageable.

- **Differences in permissions**: In the guardian relationship, what the guardian can do does not always have to be the same. These permissions may vary depending on the relationship. For example, a person can only be a guardian for a dependent's medical needs, but the dependent himself is responsible for managing other digital assets. Alternatively temporary guardianship may occur for a certain time interval or situation. Likewise, what data the guardian will present as proof of the dependent person may change.

4 Guardianship Use Case: Joint Custody

Custodianship is a relationship in which a person is legally cared for. Child custody is often the situation where the child's care and control are undertaken following a divorce. Child custody is based on an international legal basis such as *United Nations (UN) Convention on the Rights of the Child* (UN, 1989) that is accepted by the UN members and monitored by UN Committee on the Rights of the Child. The contents of this convention include issues such as the right of every child to have life, have an identity in her name and be raised by a family or cultural group. It also states that even if the parents are separate, the child should establish a relationship with both parents.

Joint custody is the situation that allows parents to exercise their rights over their underage children together and equally in a way that takes care of the child's best interests. Naturally, this is also the case in parenting. The mother and father have equal voice over the child. Also, with international agreements, joint custody decisions can be made as a result of divorces in most countries. After the divorce with joint custody, the child's parent has the right to speak equally in matters such as the child's care, protection, education, and supervision, just as before the divorce. Thus, neither the mother nor father has any advantage over each other.

That is why the Joint custody scenario differs from other guardianship scenarios. Unlike other scenarios, there are two guardians in the joint custody scenario. There will be things that cannot be done without the approval of these two guardians. Therefore, when designing the Guardianship model for this use case, it is necessary to consider in detail. The permissions and restrictions must be clearly defined in the model in order not to create vulnerabilities in the system. Sovrin Guardianship Working Group specifies three building blocks to be used in guardianship modeling as follows (Hardman, 2019):

1. **Guardianship Trust Framework:** A framework that enforces certain rules and restrictions to be applied in real-life use-cases should be used.

2. **Guardianship Credential:** Particular guardianship credentials indicating the content and the boundaries of the relationship between the actors should be used.

3. **Guardianship Challenge:** It is the structure that primarily evaluates the credentials for specific situations and controls their legality. It creates an opportunity for auditing and enforcement.

In this study, we will create the Custody Framework for the joint custody use case and publish a sample custody credential.

4.1 Main Construction

Firstly, for the framework to be used in this use case, a competent authority must create this. We can think of it as the Ministry of Justice in our example. Therefore the Ministry of Justice must define a framework to be used for custody. It has to be both machine-readable such as JSON-LD and human-readable format. This framework must be at a publicly accessible URI address. Because schema to be used to give custody credentials is defined in this framework. The institution that will issue the credential must comply with the rules contained in these schema definitions. Therefore, the framework created for this study can be accessed in Dündar (2020a) on the Github page.

In this framework, schema definition and credential definition must be defined before a credential can be issued. Schema definitions are a template that contains the information of attributes. Credential Definition has the issuer's public key data for a schema. Thus verifier can verify the signed attribute via checking the public key in the credential definition. The issuer can create the credential by filling in the attributes contained in the schema definition. In our

case, in order for the court to issue a custody credential as an issuer, firstly must have created a credential definition using the schema definition described in the custody trust framework.

Verifiable Credentials are available for expansion as specified in the implementation document. The default credential types can be used, or a new credential type can be defined. Therefore, we will use a VC type defined within our framework to be used for Custodianship.

In the Joint Custody use case, there are several actors. The characters and their stories in this use case are summarized in Table 2

Table 2: User Story Summary of Joint Custody Use Case

Persona Name	Summary	Trusted
Alice	She is a 40-year-old mother with an 8-year-old child named Charlie. She recently divorced her husband Bob.	No
Bob	Bob is 42 years old, Alice's wife and Charlie's father.	No
Charlie	Charlie is the only child of Alice and Bob, 8 years old. He goes to primary school.	Yes
Court	This is the authority who decided on joint custody for Charlie.	Yes

4.2 Implementation

In the definition of the framework, many different situations should be considered, and it should be defined in a way that does not allow weaknesses in the system. The framework started to be created with information such as the name, version, and author. The boundaries of the scope where the framework can be used are specified. The ***case-result*** attribute has been added as metadata, so this framework has been enabled to work for different custody scenarios. Afterward, three attributes were determined, indicating the reason for guardianship. These are as follows: ***kinship***, ***court-order*** *and* ***enforced***. Identity information fields were created for guardian and dependent roles, respectively, under two separate titles as ***Holder*** and ***Proxied***. ***Permissions*** and ***Constraints*** fields, which are among the most important parts of the framework, have been defined. Thanks to these areas, the guardian's permissions and restrictions are bound by a legal framework. Also, it is strongly recommended that an audit trail be produced any time a guardian performs any action.

When Alice and Bob come to court for divorce, the court will decide and create a custody credential for court results. This credential will be given on behalf of Charlie. But due to the guardian relationship, its control will be in his family. This case results in joint custody. Accordingly, the court in the role of Issuer needs to create a credential.

A sample SSI network will be created to realize the usage scenario of this credential. It will be demonstrated that the system is working properly over this network. For this purpose, the Verifiable Organization Network (VON) will be used (British Columbia, 2018). VON Network provides a ledger browser on the Docker environment. Ledger browser allows us to examine transactions and see the status of nodes locally. After that, the SSI network is ready; actors can now be created for the joint custody use case. For this, the open-source Hyperledger Aries Cloud Agent Python (ACA-Py) library will be used. Aries Cloud Agents will be also run over Docker. In this scenario, a connection will be established between the agents. Then credentials will be issued from one agent to another, and this credential be verified via *present proof* protocol.

In this study, Swagger was used to observing the endpoints of the agents easily. A Swagger is an open-source tool that uses the OpenAPI specification (swagger.io, 2020). Thanks to this tool, transactions performed on Agent endpoints can be viewed visually on the browser. First, by executing the python code we prepared for the court in the role of Issuer, we are making the Court

Agent up and running. Python code for court available on GitHub (Dündar, 2020b). Likewise, we make Alice's agent up and running. In this python script, we already registered Agents' DIDs into the public Indy ledger. The court agent published the schema and credential definition, as seen in Appendix A. It can also confirm that Schema and Credential Definition are on the ledger using the VON Network interface on the browser. Court Agent sends a credential offer to Alice's Agent. In ACA-Py, it is configured to handle credential offers automatically. Thus Alice's agent instantly responds with a credential request. The issuer receives the credential request and publishes the credential for Alice using the custody schema. This credential can be seen in Appendix B.

Thus, by implementing the framework we created for the custody scenario, we have shown that it works in an example SSI network.

5 Discussion and Analysis

In this chapter, the results obtained will be analyzed, their performance will be compared with their current systems, and their applicability will be discussed.

First, we analyzed the average time for a standard credential to issue on the current system. For this we used Python's NumPy (numpy.org, 2006) and matPlotLib (matplotlib.org, 2003) library. Thanks to this library, we were able to get the average duration of publishing guardian credentials graphically. To make an accurate measurement, we applied the *Straight Line Fitting* analysis method mentioned in Moreno & Fischmeister (2017).

To compare the Custody Framework we have created, we chose the Faber College Demo from the classic SSI samples in the ACA-Py library. First, we made 15 different measurements on Faber Demo in accordance with the Straight Line Fitting method. These measurements were made on the function that included the creation of a credential. Thus, we were able to find the time to create an average credential. We did the same for our Custody use case. As a result,

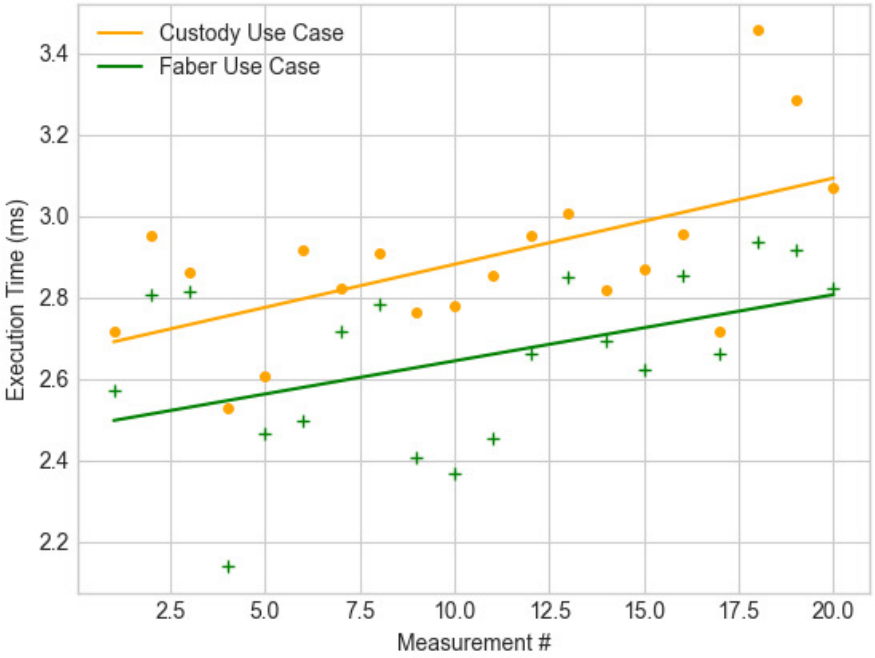

Figure 6: Credential creation time comparison between Faber Use Case and Custody Use Case

we obtained the graphic shown in Figure 6. Orange dots and green pluses show the measured results in ms. The slope of the straight lines shown in the graph gives us the average time. Accordingly, the Faber Demo's average time is $0.02ms$, while for Custody Demo, it is $0.04ms$.

The difference is an expected situation. Because guardian credentials have more restrictions than standard credentials, Credential schemas have many more attributes. The difference is negligible. The result also same for the verification process as seen in Figure 7. Thus, the result we obtained from these measurements shows that the custody framework does not bring too much extra load to existing systems.

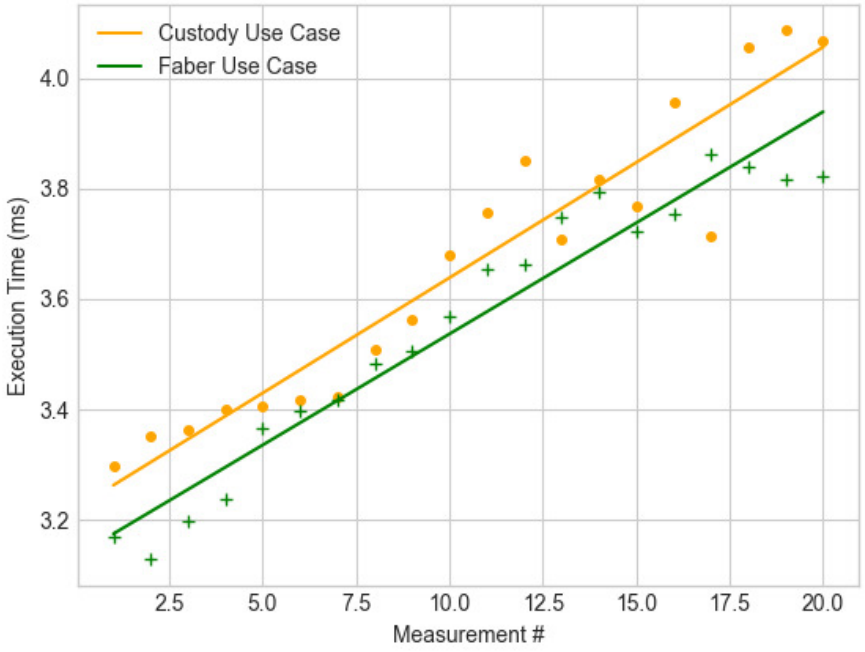

Figure 7: Verify the proof time comparison between Faber Use Case and Custody Use Case

The point to be careful here is that we took only the part of making the credential schema into a credential offer in order not to be affected by the delays caused by the network. We did not include the function of publishing this credential to the ledger on this measure. However, the measurement results, including the publication of credential to ledger, are also shown in Table 3.

Table 3: Time Comparison With Custody Credential

	Faber Use Case	**Custody Use Case**
Publishing Schema and Credential Definition complete time	5.66s	7.65s
300 credential exchanges complete time	58.10s	67.80s
Average time per credential	0.12s	0.23s

Another discussion topic is mass adoption. If SSI systems are intended to be used widely, it should meet every real-life scenario. Therefore, the guardianship is vital because many people cannot manage their digital data in real life. For the Guardian relationship to be adopted by large circles, it must be producing solutions for all kinds of Guardian-Dependent relations. Therefore, we have designed a sample system through the mutual guardian relationship scenario and proved that this system could work with the existing infrastructure. In this way, we have shown that the mutual guardian relationship scenario, which will indeed be encountered in real life, is not an obstacle in adopting SSI. One step closer to realizing mass adoption.

Many more solutions can be offered for the Joint Custody use case. The relationship between Holder-Verifier-Issuer is multilayered. Therefore, while looking for a solution to this problem,

we should consider the possible solutions separately on these different layers. However, we have shown that the process of realizing the joint custody scenario by changing the credential attributes, as described in Section 4.2, different solutions can be considered for future studies.

6 Conclusion

The model design was carried out through the Joint Custody use case. The Custody Trust framework was created for the Joint Custody use-case, and the custody credential schema was created using this framework. To ensure that this model is working on existing systems, a custody credential was created and published on a sample SSI network. Verifier has approved the credential's present proof, thus ensuring that the operation works properly within the existing SSI structure. After making sure that the Custody Framework created for mutual guardianship is working correctly, performance analysis is performed with existing systems, and the results are added to the study with graphics.

As a result, in this study, the topic of guardianship was taken one step forward under the guidance of the studies made by the Guardianship Working group under the Sovrin Foundation and a scenario where one person's digital identity could be managed jointly by two people was studied. An example of this situation in real life is when a couple gets joint custody of their children as a result of divorce. In this case, what kind of differences should be done in digital identities' infrastructure has been researched, a model has been created and a new way about mutual guardianship has been opened for further studies.

7 Acknowledgement

The authors would like to thank anonymous reviewers for insight-full feedback and comments.

References

Abraham, A. (2017). *Self-sovereign identity.* Styria: E-Government Innovationszentrum.

Adkins, H. (2011). An update on attempted man-in-the-middle attacks. *Google Security Blog* https://security.googleblog.com/2011/08/update-on-attempted-man-in-middle.html

Ali, M., Nelson, J., Shea R. & Freedman, M. J. (2016). Blockstack: A global naming and storage system secured by blockchains, *2016 Annual Technical Conference 16*, 181–194.

Allen, C. (2016). The Path to Self-Sovereign Identity. http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html

Allen, C. & Appelcline S. (2019). SmartCustody: Use of Advanced Cryptographic Tools to Improve the Care, Maintenance, Control, and Protection of Digital Assets. *Smart Custody* https://www.smartcustody.com/

Assembly, UN General. (1989). Convention on the Rights of the Child. *United Nations, Treaty Series 1577*(3).

Camenisch, J., Drijvers, M. & Dubovitskaya, M. (2017). Practical UC-secure delegatable credentials with attributes and their application to blockchain. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 683–699.

Camenisch, J. & Van Herreweghen, E. (2002). Design and implementation of the idemix anonymous credential system. *Proceedings of the 9th ACM conference on Computer and communications security*, 21–30.

Civic, C. (2017). Civic - Our Mission. https://www.civic.com/company/

Davie, M., Gisolfi, D., Hardman, D., Jordan, J., O'Donnell, D. & Reed, D. (2019). The Trust over IP Stack. *IEEE Communications Standards Magazine 3*(4), 46–51.

Der, U., Jähnichen, S. & Sürmeli, J. (2017). Self-sovereign Identity − Opportunities and Challenges for the Digital Revolution. *arXiv preprint arXiv:1712.01767*.

Doerk, A. & Helper H. (2019). An introduction to self-sovereign identity. (SSI). *Medium* https://ssi-ambassador.medium.com/an-introduction-to-self-sovereign-identity-ssi-916eb42f0490

Dündar, Y. (2020a). 2020-MScThesis-Custody Trust Framework. https://github.com/yusufdundar/2020-MScThesis

Dündar, Y. (2020b). 2020-MScThesis-Custody Trust Framework Source Code. https://github.com/yusufdundar/2020-MScThesis/tree/master/src

Dunphy, P., Garratt, L. & Petitcolas, F. (2018). Decentralizing Digital Identity: Open Challenges for Distributed Ledgers. *2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, 75–78.

George, N. (2019a). Announcing Hyperledger Aries, infrastructure supporting interoperable identity solutions. *Hyperledger* https://www.hyperledger.org/blog/2019/05/14/announcing-hyperledger-aries-infrastructure-supporting-interoperable-identity-solutions

George, N. (2019b). "Hyperledger Aries: Open Source Interoperable Identity Solution" *SSI Meetup* https://ssimeetup.org/hyperledger-aries-open-source-interoperable-identity-solutions-nathan-george-webinar-30/

Gibbs, S. (2016). Dropbox hack leads to leaking of 68m user passwords on the internet. *The Guardian* https://www.theguardian.com/technology/2016/aug/31/dropbox-hack-passwords-68m-data-breach

Hackett, R. (2016). LinkedIn Lost 167 Million Account Credentials in Data Breach. *Fortune* https://fortune.com/2016/05/18/linkedin-data-breach-email-password/

Hardman, D. (2019). Aries RFC 0103: Indirect Identity Control. https://github.com/hyperledger/aries-rfcs/tree/master/concepts/0103-indirect-identity-control

Hardt, D. et al. (2012). The oAuth 2.0 authorization framework, *Technical Report RFC 6749*.

Hyperledger (2018). Hyperledger Indy Documentation v1.0a. https://hyperledger-indy.readthedocs.io/en/latest/index.html

Hyperledger (2019a). Hyperledger Aries Cloud Agent - Python. https://github.com/hyperledger/aries-cloudagent-python

Hyperledger (2019b). Sample Guardianship Trust Framework. https://github.com/hyperledger/aries-rfcs/blob/master/concepts/0103-indirect-identity-control/guardianship-sample/trust-framework.md

id2020.org (2018). The Alliance Manifesto. https://id2020.org/manifesto

Kemp, S. (2019). Digital 2019: Global Digital Overview. *Data Reportal* https://datareportal.com/reports/digital-2019-global-digital-overview

Kondova, G. & Erbguth J. (2020). Self-sovereign identity on public blockchains and the GDPR. *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, 342–345.

Lodder, M., Zundel, B. & Khovratovich, D. (2019). Pairings-based Anonymous Credentials with Circuit-based Revocation and Permission Policies.

Lundkvist, C., Heck, R., Torstensson, J., Mitton, Z. & Sena, M. (2017). Uport: A platform for self-sovereign identity. *UPort*, https://whitepaper.uport.me/uPort_whitepaper_DRAFT20170221.pdf

Maler, E., Cahill, C., Hughes, J., Beach, M. & Metz, R.B et al. (2005). Security and privacy considerations for the oasis security assertion markup language (saml) v2.0. *Language (SAML)(2)*.

Matplotlib.org. (2003). *Matplotlib: Visualization with Python*, https://matplotlib.org

Moreno, C. & Fischmeister S. (2017). Accurate Measurement of Small Execution Times Getting Around Measurement Errors. *IEEE Embedded Systems Letters 9(1)*, 17–20.

Nakamoto, S. (2008). A peer-to-peer electronic cash system. *Bitcoin*, https://bitcoin.org/bitcoin.pdf

Natarajan, Krause, H., Gradstein, S. & Helen. (2017). Distributed Ledger Technology and Blockchain *World Bank, doi:10.1596/29053*.

Numpy.org. (2006). *The fundamental package for scientific computing with Python*, https://numpy.org

Office of the United Nations High Commissioner for Human Rights (1948). Universal Declaration of Human Rights. http://www.unhchr.ch/udhr/lang/eng.pdf

Project Liberty Alliance (2005). Liberty ID-FF Architecture Overview Version: 1.2-errata-v1.0. http://www.projectliberty.org/liberty/content/download/318/2366/file/draft-liberty-idff-arch-overview-1.2-errata-v1.0.pdf

Province of British Columbia (2017). "OrgBook BC" https://orgbook.gov.bc.ca/en/home

Province of British Columbia (2018). "VON Network" https://github.com/bcgov/von-network

Reed, D., Law, J. & Hardman, D. (2016). The Technical Foundations of Sovrin. *Evernym*, https://www.evernym.com/wp-content/uploads/2017/07/The-Technical-Foundations-of-Sovrin.pdf

Reed, D., Sporny, M., Longley, D., Allen, C., Grant, Ryan & Sabadello, M. (2020). Decentralized Identifiers (DIDs) v1.0 *World Wide Web Consortium*, https://www.w3.org/TR/did-core/

Rountree, D. (2012). Federated identity primer. *Newnes*.

Ruff, T. (2020). Introducing Self-Sovereign Student ID. *Medium*, https://rufftimo.medium.com/introducing-self-sovereign-student-id-part-1-of-2-d3104ab0f8f2

Sakimura, N., Bradley, J., Jones, M., Medeiros, B.& Mortimore, C. (2014). OpenID Connect Core 1.0. *OpenID Foundation*, https://openid.net/specs/

Security.org. (2018). *Public Awareness of Major Data Breaches*, https://www.security.org/resources/data-breach-awareness/

Sporny, M., Longley, D. & Chadwick, D. (2019). Verifiable Credentials Data Model 1.0. *World Wide Web Consortium*, https://www.w3.org/TR/vc-data-model/

Sovrin Foundation (2018). *What is self-sovereign Identity?*, https://sovrin.org/faq/what-is-self-sovereign-identity/

Sovrin Foundation (2019a). *Sovrin Glossary V3*,

Sovrin Foundation (2019b). *Sovrin Governance Framework*, https://sovrin.org/library/sovrin-governance-framework/

Sovrin Foundation (2019c). *Sovrin Governance Framework V2*, https://sovrin.org/wp-content/uploads/Sovrin-Governance-Framework-V2-Master-Document-V2.pdf

Sovrin Guardianship Task Force (2019). *On Guardianship in Self-Sovereign Identity*, https://sovrin.org/wp-content/uploads/Guardianship-Whitepaper2.pdf

Spangler, T. (2017). *Yahoo Says 3 Billion User Accounts Were Hacked, Upping Previous Estimate*, https://finance.yahoo.com/news/yahoo-says-3-billion-user-205330487.html

Swagger.io. (2020). *OpenAPI Specification*, https://swagger.io/specification/

The Linux Foundation. (2018). "Hyperledger Ursa" https://www.hyperledger.org/use/ursa

Tobin, A. & Reed D. (2016). The inevitable rise of self-sovereign identity. *The Sovrin Foundation (29)*.

Van Bokkem, D., Hageman, R., Koning, G., Nguyen, L. & Zarin, N. (2019). Self-sovereign identity solutions: The necessity of blockchain technology. *arXiv preprint arXiv:1904.12816*.

van Wingerde, M. (2017). Blockchain-enabled self-sovereign identity. *Master's thesis, Tilburg University, School of Economics and Management*.

Weller, D. & Dijksman R. (2019). Blockchain's Relationship with Sovrin for Digital Self-Sovereign Identities.

Windley, P. (2016). How Sovrin Works. *Windley*, https://www.windley.com/archives/2016/10/how_sovrin_works.shtml

Windley, P. (2017). Fixing the Five Problems of Internet Identity. *Windley*. https://www.windley.com/archives/2017/10/fixing_the_five_problems_of_internet_identity.shtml

World Bank (2018). Identification for Development (ID4D) Global Dataset. *World Bank*. https://datacatalog.worldbank.org/dataset/identification-development-global-dataset

Appendix A Sample Custody Schema

```
Startup duration: 4.04s
Admin URL is at: http://192.168.65.3:8021
Endpoint URL is at: http://192.168.65.3:8020

#3/4 Create a new schema/cred def on the ledger
Schema:
  {
    "schema_id": "Mknh6qTtuiyykRe63wMQ3q:2:custody schema:96.53.21",
    "schema": {
      "ver": "1.0",
      "id": "Mknh6qTtuiyykRe63wMQ3q:2:custody schema:96.53.21",
      "name": "custody schema",
      "version": "96.53.21",
      "attrNames": [
        "credentialSubject.holder.type",
        "credentialSubject.holder.constraints.boundaries",
        "credentialSubject.proxied.iris",
        "trustFrameworkURI",
        "issuanceDate",
        "credentialSubject.proxied.fingerprint",
        "appealURI",
        "credentialSubject.holder.rationaleURI",
        "credentialStatus",
        "credentialSubject.proxied.firstName",
        "credentialSubject.proxied.lastName",
        "credentialSubject.holder.constraints.jurisdictions",
        "id",
        "credentialSubject.holder.constraints.startTime",
        "credentialSubject.holder.constraints.circumstances",
        "type",
        "credentialSubject.holder.permissions",
        "expirationDate",
        "credentialSubject.holder.kinshipStatus",
        "credentialSubject.proxied.type",
        "caseResult",
        "credentialSubject.holder.role",
        "credentialSubject.proxied.photo",
        "credentialSubject.holder.lastName",
        "credentialSchema",
        "credentialSubject.holder.firstName",
        "credentialSubject.holder.constraints.endTime",
        "credentialSubject.holder.constraints.radiusKM",
        "credentialSubject.proxied.birthDate",
        "credentialSubject.holder.constraints.pointOfOrigin",
        "credentialSubject.holder.constraints.trigger",
        "auditURI",
        "issuer"
      ],
      "seqNo": 16
    }
  }

Schema ID: Mknh6qTtuiyykRe63wMQ3q:2:custody schema:96.53.21
Cred def ID: Mknh6qTtuiyykRe63wMQ3q:3:CL:16:default
Publish schema/cred def duration: 9.99s
```

Figure 8: Court Agent's Schema for Custody

Appendix B Sample Custody Credential


```
Credential details:
  {
    "referent": "5d294c1c-4fa3-4710-8952-6505e39a8d1d",
    "attrs": {
      "credentialSubject.proxied.birthDate": "2007-07-01",
      "caseResult": "joint-custody",
      "credentialSubject.holder.constraints.circumstances": "|en: While a parent or adult sibling is unavailable, and no new guardian has been adjudicated",
      "credentialSubject.holder.constraints.jurisdictions": "tur",
      "credentialSubject.holder.rationaleURI": "court-order",
      "credentialSubject.proxied.fingerprint": "null",
      "credentialSubject.holder.type": "Holder",
      "credentialSubject.proxied.firstName": "Charlie",
      "credentialSubject.holder.role": "kinship",
      "credentialSubject.holder.lastName": "Smith",
      "trustFrameworkURI": "https://github.com/yusufdundar/2020-MScThesis/blob/master/custody-framework.md",
      "issuanceDate": "1593713286",
      "credentialSubject.holder.constraints.startTime": "2007-07-01 T18:00",
      "credentialSubject.holder.kinshipStatus": "mother",
      "appealURI": "https://example.org/appeal",
      "credentialSubject.holder.constraints.radiusKM": "1000",
      "credentialSubject.proxied.iris": "null",
      "credentialSubject.holder.constraints.pointOfOrigin": "Ankara",
      "credentialSubject.holder.permissions": "routine-medical-care",
      "credentialSubject.holder.firstName": "Alice",
      "credentialSubject.proxied.lastName": "Smith",
      "credentialSubject.holder.constraints.boundaries": "Within  Turkey  country  limits",
      "credentialSubject.proxied.type": "Proxied",
      "credentialSubject.holder.constraints.endTime": "2025-08-01",
      "issuer": "https://moj.gov/issuers/14",
      "auditURI": "https://example.org/audit",
      "credentialSubject.proxied.photo": "https://raw.githubusercontent.com/yusufdundar/2020-MScThesis/master/charlie.base64",
      "credentialSubject.holder.constraints.trigger": "|en: Death of  parent"
    },
    "schema_id": "NJiYGdiNhdm8ZCfedpNCzM:2:custody schema:57.24.93",
    "cred_def_id": "NJiYGdiNhdm8ZCfedpNCzM:3:CL:40:default",
    "rev_reg_id": null,
    "cred_rev_id": null
  }

Credential request metadata:
  {
    "master_secret_blinding_data": {
      "v_prime": "42038558799018855740133663747148856923173233228163099647287684958824673059021361552303484052638014179026855791681990525308118392198991420103350136329176491963827010888330809791533879902727885500927751185087207256598321185949719087370686758829494074984403782566311705700226546028874694053366838943343941564604799983221154456654238244024965085073774215207098170565325270147815337861623562835756474821098502595299885425479895844803547964312974056946610616798749857906004637232885561735132877563164881481921712722947857498425185110670542592377301060111628948805420168519511640331056079993292934502599005143034944811820274300708679793438819400 19",
      "vr_prime": null
    },
    "nonce": "24637291029103025512024 2",
    "master_secret_name": "alice.agent307475"
  }

Alice      | credential_id 5d294c1c-4fa3-4710-8952-6505e39a8d1d
Alice      | credential_definition_id NJiYGdiNhdm8ZCfedpNCzM:3:CL:40:default
Alice      | schema_id NJiYGdiNhdm8ZCfedpNCzM:2:custody schema:57.24.93
```


Figure 9: Credential Issued by Court Agent