MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES
IN PURE AND APPLIED SCIENCES

ON SELF SOVEREIGN IDENTITY BASED GUARDIANSHIP

YUSUF DÜNDAR

MASTER THESIS
Department of Cyber Security

Thesis Supervisor
Dr. Hüseyin YÜCE

Thesis CO- Supervisor
Dr. İsa SERTKAYA

ISTANBUL, 2021

Acknowledgments

I would like to express my gratitude to my thesis co-advisor Dr. İsa Sertkaya for their guidance and constant encouragement. I am very grateful for scientific advice, knowledge and many insightful discussions and suggestions. It has been an honor to work with him.

I would like to thank to my thesis committee: Dr. Hüseyin Yüce, Assoc.Prof.Dr.Önder Demir and Dr.Mustafa Cem Kasapbaşı for their encouragement, insightful comments and questions.

I would also thank to TÜBİTAK BİLGEM for its opportunity to study this Masters Degree and contribution on my experience on Distributed Identity Management Systems.

A very special thanks goes to my dear family and my friends who motivated and encouraged me to finish this thesis.

Contents

Acknowledgments **i**

Abstract **iv**

Öz **v**

Abbreviations **vi**

List of Figures **vii**

List of Tables **viii**

1 Introduction **1**
1.1 Identity Management Scheme Types 2
1.2 Related Work 10
1.3 Our Contributions 11
1.4 Organization 11

2 Preliminaries **13**
2.1 Cryptographic Primitives 13
2.1.1 Digital Signatures 13
2.1.2 Hash Functions 14
2.1.3 Camenish Lyanskaya (CL) Signatures 14
2.1.4 Zero Knowledge Proof 15
2.2 Verifiable Credentials 15
2.2.1 Decentralized Identifiers (DIDs) 18
2.2.2 Anonymous Credentials 20
2.3 Distributed Ledger Technologies 22
2.4 Self Sovereign Identity 24
2.4.1 Sovrin Ledger 24
2.4.2 Sovrin Agents 26
2.4.3 Sovrin Clients 27
2.4.4 Sovrin Infrastructure 28
2.4.5 Sovrin SSI Credential Management 30

3 Indirect SSI Control: Guardianship **34**
3.1 Guardianship 37
3.2 Guardianship Life Cycle 40
3.3 Indirect SSI Control Risks and Challenges 41

3.3.1 Risks arise within Guardianship 41
3.3.2 Challenges 43
3.4 Guardianship Conceptual Model 44
3.5 Proof of Concept Use Case: Refugee Camp 47

4 Guardianship Use Case: Custody **53**
4.1 Joint Custody 54
4.1.1 Formalization of Joint Custody 55
4.2 Proof of Concept Use Case: Joint Custody 56
4.2.1 Main Construction 57
4.2.2 Implementation 60

5 Discussion and Analysis **64**

6 Conclusion **67**

A Credential Schema Example **70**

B Custody Schema **72**

C Custody Trust Framework **74**

D Custody Credential **77**

E Credential Issued by Court Agent **81**

Bibliography **82**

ON SELF SOVEREIGN IDENTITY BASED GUARDIANSHIP

Yusuf Dündar

Abstract

Self Sovereign Identity (SSI) is a new approach that develops promising solutions on the longstanding Identity problem. While the digital identities are increasing day by day, the physical Identities such as passport, driver's license, have started to be digitalized thanks to new technologies. With these two worlds getting more intertwined, a new identity management system was needed for identities containing the most critical information. SSI comes into play in this regard and proposes a new model with certain principles such as privacy, Interoperability, and consent.

SSI allows the person to manage the identity data without a 3rd party trusted institution. The person is solely responsible for her own identity data. She decides whom to share with, what information to share. It realizes this by using technologies such as Blockchain and Verifiable Credential in its infrastructure. For the widespread use of such solutions, solutions already used in daily life should also be considered. In this context, indirect SSI control methods are examined in this study, and proof of concept solutions are made over real life scenarios. In particular, the guardianship that occurs when the person becomes unable to manage their own identity is very important for the mass adoption of SSI. Several Working Groups continue their work on guardianship. In the current studies, an exemplary guardianship trust framework has been created on a real life refugee girl scenario.

In this study, a guardianship framework was designed based on real life custodianship scenario. The reason for choosing this scenario is that in cases such as joint custody, one person has two guardians; therefore, management of the dependent's identity must be jointly carried out. A proof of concept was made for the Custody Framework, which overcame this problem. Modeling of the framework was carried out on a sample SSI network, and performance analysis was performed with existing systems.

As a result, it has been shown that the designed framework can work without overloading the existing systems.

Keywords: self sovereign identity, ssi, indirect ssi control, privacy, guardianship, custodianship

KULLANICI EGEMEN KİMLİK TABANLI VASİLİK

Yusuf Dündar

Öz

Kullanıcı Egemen Kimlik (SSI), uzun süredir devam eden kimlik problemine umut vadeden çözümler getiren yeni bir yaklaşımdır. Dijital kimliklerin kullanımı günden güne artarken, ehliyet ve pasaport gibi fiziksel kimlikler ise yeni teknolojiler sayesinde dijitalleşmeye başlamıştır. Bu iki dünyanın gitgide iç içe geçmesi ile yeni bir kimlik yönetim sistemi ihtiyacı doğdu. SSI tam da bu konuda, mahremiyet, müşterek çalışma, rıza gibi prensipleri göz önünde bulundurarak yeni bir model sunmaktadır.

SSI 3.parti güvenilir bir kuruluşa gerek kalmadan kişinin kendi kimlik verilerini yönetmesine izin verir. Kişi bu verilerinin kimlerle ve ne kadarının paylaşılacağından kendisi sorumludur. Bunu da altyapısında blockchain ve VC gibi teknolojileri kullanarak gerçekleştirir. Bu tür çözümlerin yaygın olarak kullanılması için mevcut çözümlerinde göz önünde bulundurulmasıgerekir. Bu kapsamda dolaylı SSI kontrolü yöntemleri bu çalışmada incelenmiş ve gerçek hayat senaryolarıüzerinden kavram ispatlama yapılmıştır. özellikle guardianship kavramı SSI çözümünün kitlesel benimsemesi için oldukça önemlidir. Mevcut çalışmalarda belirli çalışma grupları guardianship üzerine araştırmalarına devam etmektedir.

Bu çalışmada gerçek hayattan velayet senaryosu baz alınarak guardianship çatısı tasarlanmıştır. Bu senaryonun seçilmesinin sebebi, ortak velayet gibi senaryolarda gerçekleşecek olan bir kişinin iki vasisi olma durumunu ihtiva etmesidir. Bu durumda bu iki kişinin ortaklaşa olarak vasinin dijital kimliğini yönetmesi söz konusu olacaktır. Bu problemi üstesinden gelen Custody Frameworkünün kavram ispatlaması yapılmış, örnek bir SSI networkü üzerinde modellenmiş ve mevcut sistemler ile performans analizi yapılmıştır.

Sonuç olarak tasarlanan Custody Frameworkünün ekstra yük getirmeden mevcut sistemler ile uyumlu bir şekilde çalıştığı gösterilmiştir.

Anahtar Sözcükler: kulanıcı egemen kimlik, ssi, dolaylı ssi kontrolü, mahremiyet, vasilik, vekalet

Abbreviations

ACA-Py	**A**ries **C**loud **A**gent **Py**thon
CA	**C**ertificate **A**uthority
CL	**C**amenish **L**yanskaya
DID	**D**ecentralized **Id**entifier
DKMS	**D**istributed **K**ey **M**anagement **S**ystem
DLT	**D**istributed **L**edger **T**echnologies
IDP	**ID**entity **P**rovider
IoT	**I**nternet **O**f **T**hings
PKI	**P**ublic **K**ey **I**nfrastructure
RBFT	**R**edundant **B**yzantine **F**ault **T**olerant
SGF	**S**ovrin **G**overnance **F**ramework
SGFWG	**S**ovrin **G**overnance **F**ramework **W**orking **G**roup
SGL	**S**imple **G**rant **L**anguage
SGWG	**S**ovrin **G**uardianship **W**orking **G**roup
SPTF	**S**ovrin **P**rovisional **T**rust **F**ramework
SSI	**S**elf **S**overeign **I**dentity
ToIP	**T**rust **o**ver **IP**
UN	**U**nited **N**ations
UUID	**U**niversally **U**nique **ID**entifier
VC	**V**erifiable **C**redential
VP	**V**erifiable **P**resentation
VON	**V**erifiable **O**rganizations **N**etwork
W3C	**W**orld **W**ide **W**eb **C**onsortium
ZKP	**Z**ero **K**nowledge **P**roof

List of Figures

1.1 Self Sovereign Identity Actors [1]. 7

2.1 The roles and information flows in Verifiable Credentials [2] 16
2.2 Life of Single Verifiable Credential [2] 17
2.3 Syntax of Decentralized Identifiers [3] 19
2.4 Generic structure of a blockchain [4] 22
2.5 The two basic node types in the Sovrin ledger [5] 25
2.6 Sovrin agents of the Sovrin network [5] 26
2.7 The four layers of the Trust over IP stack [6] 28
2.8 Hyperledger Aries Architecture [7] 30
2.9 Digital wallets for guardian and dependent [8] 31
2.10 Self Sovereign Identity Architecture [1]. 32

3.1 The three types of indirect identity control relationships [8] 36
3.2 Transparent vs. opaque guardianship [8] 39
3.3 The four major stages in the guardianship life cycle [8] 41
3.4 Guardianship type field notation [9] 46
3.5 Mya User Story - Identity Needs [10] 48

4.1 List of containers running on Docker 61
4.2 Nodes status screen over local VON Network 62
4.3 Court Agent Swagger User Interface on Browser 62
4.4 Court Agent's Schema for Custody 63

5.1 Comparison between Faber Use Case and Custody Use Case 65
5.2 Verify the proof time comparison between Faber Use Case and Custody Use Case 66

E.1 Credential Issued by Court Agent 81

List of Tables

1.1 Ten Principles of Self Sovereign Identity summarised by the Sovrin Foundation [11] 6

2.1 Comparison of DIDs with URLs and Email Addresses [12] 20

3.1 Primary distinctions between the three types of identity control relationships [13] 37

4.1 User Story Summary of Our Use Case 56

5.1 Time Comparison With Custody Credential 65

Chapter 1

Introduction

Identity is defined as "*The set of characteristics that make you who you are.*" in the [14]. In order to understand the identity concept in detail, one needs further clarification of the realm and context where the identity is used.

In real life, identity is much more than a name or mail address. It reflects one's personality or credentials and is used for authentication. One can prove via credentials issued by central authorities such as national Id cards, passports, and driver licenses. Even if these credentials uniquely identify someone, they may be used in certain situations such as work-persona or family-persona. However this does not mean that we have several identities. These different identity components such as physical appearance, behaviour, likes or dislikes, or personal preferences come together to create our real life identity.

Digital identity, on the other hand, is used in an extensive concept and involves digital characteristics, digital behavior, pieces of physical identity, and personal information. As stated in [15], the digital identity contains data that uniquely describes a person (subject) or thing (entity) but also has information about the subject's relationships with other entities. This statement is essential because this relationship between entities requires trust in the online world. This leads to the ongoing problem of current digital identity systems, which is the "Trust Problem".

Before we dig into these trust issues of the identity systems, we first review the evolvement of identity management scheme types.

1.1 Identity Management Scheme Types

Since digital identities first appeared, they have gone through four main stages to evolve today's systems. Christopher Allen defines four stages of digital identity in [16]. We have examined these stages in detail below.

- **Centralized Identity (administrative control by a single authority or hierarchy)** 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. But still, there are core authorities in the root that holds up the power. As the Internet grows, problems started to occur. The identities that users have to manage are increasing. The real trouble is that the users have no control over these identities. The need for interoperability between the different website identities caused Federated Identity to appear.

- **Federated Identity (administrative control by multiple, federated authorities)** The next big step up for digital identities began with the central authorities decide to eliminate the incompatibilities between them. Federated identity is described as

  > "The set of agreements, standards and technologies that enable a group of service providers to recognise user identifiers and entitlements from other service providers within a federated domain."

  in [17].

  Federated Identity relies on Identity Providers (IdP). IdPs are the system component that creates and manage the identity information. User first required to register with an IdP. Then the user can authenticate an application with these IdP-owned identities. The application does not perform any actions to verify the user's identity. Because there is a trust relationship between the application and IdP, thus, application trusts the verification of identity coming from IdP. A single identity is sufficient for the user to access all services in the federated domain.

  SAML [18] and LAP [19] are the open-source projects that work on the federated identity model.

- **User-Centric Identity (individual or administrative control across multiple authorities without requiring a federation)** Federated Identity was a step forward from a Centralized Identity. It addresses the problem of users are having dozens of IDs. But still, it is a centralized system and administrated by federated commercial companies. Therefore communities begin to work on solutions that the users are in the center of the process, not the IdPs. Internet Identity Workshop working group [20] focused on a new term that countered the server-centric model of centralized authorities. Thereby user-centric identity notion was born. User-centric methodologies tend to work on two elements: user consent and interoperability [16]. Several initiatives, like OpenID Connect [21] provide more simplicity for individuals by providing Single Sign-On (SSO) functionality. OAuth [22] is an open source standard that allow secure authorization.

- **Self Sovereign Identity (individual control across any number of authorities)** Self Sovereign Identity (SSI) is a step forward from User-Centric Identity. Furthermore, it could be the solution to the trust 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 [23] by Sovrin Foundation, one of the leading non-profit organization in SSI. The fundamental principle of SSI is that people have full control over their identity data. Self sovereign identity solutions promise more privacy-enhanced identity solutions since the identity owner possesses and rules his/her credentials. To achieve decentralization, SSI solutions mostly relies on blockchain technology. Several platforms provide Self Sovereign Identity solutions such as BlockStack [24], uPort [25] and Sovrin [5].

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 [26].

- **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 certain organizations such as Service Providers create privacy problems related to our identity. These digital data become an open target for cyber attacks. At the same time, the provision of digital identity services through individual institutions constitutes Single Point of Failure.
- **The Consent Problem:** Under no circumstances should digital identity data be shared without the consent of the person.

In the early years of the Internet, the primary mechanism for authenticating yourself on an online environment is the user ID and password combination. Before, each service required the user to register with the service providers and later authenticated herself with the credentials acquired during the registration phase. This escalated very quickly and resulted in identity or credentials thefts. News about identity data breaches has been increasing in recent years. %27 of 1107 people surveyed faced data breaches in 2018, according to [27]. Prominent examples of big identity thefts involve big companies such as Yahoo [28], Dropbox [29] and LinkedIn [30].

Later federated and user-centric identity solutions came into play. It brought many advantages. For instance, with the help of Single Sign-On, users do not need to memorize/manage different identities. So it reduces the number of usernames and passwords. Also, users are no longer the give identity information every different website thus enhanced the security of the user credential. However there are also disadvantages. Firstly the federated system will just lock you up in their ecosystem. The consequences are that the users are not in control, data breaches and identity fraud are prevailing issues. Second, it requires a specialized infrastructure that cost money and time. In order to IdPs work together, they must achieve interoperability. So they need to conform to the same standard. Lastly is the "one key to the kingdom" theory. This theory explained in [14] as below.

> "The idea is that if all your applications are protected by one set of credentials and that set of credentials is compromised, then the attacker would have access to all your applications. But if all your applications are credentialed separately, then the compromise of one password would only affect one application."

We mentioned earlier the trust problem on the Internet. But what is Trust? Trust is described in [15]

> "It is a firm belief in the veracity, good faith, and honesty of another party, with respect to a transaction that involves some risk."

In the traditional centralized systems, it is hard to establish trust between authorities. It always has a risk that some trusted authorities could compromise. If the trusted authorities are compromised in some way, those identity data can be used in malicious 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 [31].

In recent years, with the rise of blockchain technology, there are new opportunities for the 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 [32].

These lead to our main subject, Self Sovereign Identity. The word sovereign in the identity system appears on the first time in [33] in 2012. Since that year, SSI has evolved with the rise of DLT and blockchain technology.

Self Sovereign Identity

Nevertheless, SSI is described in many different ways; it has certain principles to follow. These guiding principles provide better perceive the definition of SSI. Christopher Allen stated these ten key principles of self sovereign identity as follows [16].

- **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.

TABLE 1.1: Ten Principles of Self Sovereign Identity summarised by the Sovrin Foundation [11]

SECURITY	CONTROLLABILITY	PORTABILITY
Protection	Existence	Interoperability
Persistence	Control	Transparency
Minimisation	Consent	Access
		Portability

SSI solutions should realize the principles given above. These principles are crucial in order to overcome the shortcoming features of currently widely used identity management systems. With the help of these principles, many different working groups, foundation or company are trying to make SSI real-life implementation. uPort [25], BlockStack [24] and Sovrin [5] 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 critical aspects of this infrastructure is the "Verifiable Credentials (VC)". All the SSI solutions utilize World Wide Web Consortium (W3C) verifiable credentials data model [2]. W3C verifiable credentials working group defines verifiable credentials as

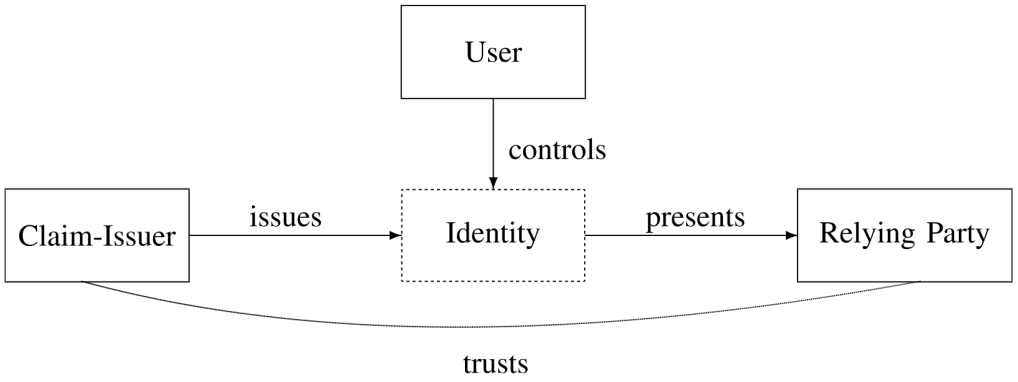


FIGURE 1.1: Self Sovereign Identity Actors [1].

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

in [2]. 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 degree, passport etc. These all help the verify that we have some attributes. A digital credential is an equivalent these credentials in the online environment. VC might consist of information related to the subject, credential type, issuing authority and some constraints same as physical credentials. But apart from this information, VC has additional attribute such as digital signatures [2]. Thereby this makes VCs cryptographically secure, privacy-enhanced and more trustworthy.

Other aspect of the infrastructure is the blockchain. Blockchain was describe in [4] as

> "It is a peer-to-peer, distributed ledger that is cryptographically-secure, append-only, immutable, and updateable only via consensus or agreement among peers."

Blockchain technology becomes more popular in recent years with cryptocurrencies. Bitcoin was introduced with the paper *"Bitcoin: A Peer-To-Peer Electronic Cash System"* by Satashi Nakamota in 2008 [34]. Since that day, many digital payment application 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 used in different areas such as art, government, big data, digital identity verification and distributed apps.

SSI mostly 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 [35]. 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 [36]. 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 [37] and Hyperledger Aries [38]. Sovrin Foundation established under the company Evernym provides real life running application and use case using their Sovrin SSI Framework and Hyperledger technologies [23]. 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) [39]. 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 [40]. 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 [2] and Decentralized Identifiers [32] continue with the several working groups.

Even if SSI has promising features such as user's ownership on his/her identity, privacy enabled authentication, selective disclosure, these features bring along 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 be able to control

their own data? For instance; older people or underage children who cannot be able to use a digital wallet to manage their credentials. To solve this problem, we have to examine the identity control mechanism in SSI systems. Firstly this mechanism must provide the following needs: [8]

- Work with existence identity management system
- Support legal, social and organizational processes
- Support revocation

In Sovrin Network, there are defined three different indirect identity control system for solving the problem mentioned above [8]. These can be shortly defined as follows and will be further studied in Chapter 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 the control of another Entity

These relationship models help manage credentials on different use cases. Guardianship provides all of these needs mentioned above. Thus it is an essential part of the Sovrin Framework. This study focused on the guardianship model.
Sovrin Foundation established "Sovrin Guardianship Working Group (SGWG)" in 2019. In which situation that the guardianship should be needed to explain in Sovrin Guardianship Whitepaper [8] as follows;

> "An Individual who does not have the capability to directly control that Individual's Identity Data (a Dependent) shall have the right to appoint another Identity Owner who has that capability (an Independent or an Organization) to serve as the owner's Guardian."

Let us exemplify these types of identity management through real-life scenarios. First of all, for the guardian relationship, the family that manages the identity of a small child who is not capable of managing their digital identity can be given as an example. In this relationship, the child has the role of a dependent. Her family has the role of guardian. For the delegation relationship, for example, we can show that the family delegates a teacher to manage specific credentials for the digital assets of the child in school. Finally, for the controller relationship, the relationships between people and things which will never be self-sovereign can be given as an example. For instance, the relationship between smart devices and their owners, or the relationship between pets and their owners.

1.2 Related Work

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

In [41], Paul Dunphy from Cambridge describe the challenges for Distributed Ledger systems. In [42], Uwe Der explores SSI systems opportunities and challenges briefly. With the rise of distributed technology, many SSI solutions came out. In [43], [44] and [45], SSI solutions such as Sovrin, Blockstack, Multichain and uPort compared in terms of, blockchain types, key management system and Proof-Of-Work system. uPort [25] is a 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 doesn't allow the users revoke their consent. Similar to uPort, same year Civic Wallet came out. Their mission to give people control on their crypto currency data [46]. Apart from digital currency related solutions, there are consortium-based solutions about identity management such as id2020 [47]. Their mission to give an identity who doesn't have an identity in the world using blockchain technologies. Also there are several works about identity. In [48], Timothy Ruff propose a Self-Sovereign Student ID. Smart Custody proposed by Christopher Allen with the aim of protecting the personal digital assets using cold storage [49]. Sovrin is a foundation based SSI solution using permissioned blockchain. It was proposed in 2016 [11]. 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 [50] and [5], it was mentioned technical details about Sovrin. It is based on Hyperledger Indy [36] and Hyperledger Aries [38] technology. It also use Decentralized Identifiers [32], idemix Anonymous Credential [51] and Verifiable Credentials [2] as a underlying technologies.

There are more SSI studies on Sovrin Network. Such as in [52], Galia Kondova studied compatibleness of Sovrin network with the EU GDPR law. In [53], Daan Weller analysed the Web of Trust and compare 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.* [54] present the first delegatable anonymous credential system that is practical in 2017. Under the Sovrin foundation, Guardianship Working Group was established in December 2019 [8]. They have published first study in [8] to form the basis for further work. They studied two real world cases. But there are alot of other use case needs to be studied. Still robust final solution has not yet been revealed regarding the guardianship.

1.3 Our Contributions

In consideration of the studies mentioned in Section 1.2, 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. Lastly, we analyzed the solution we applied in detail.

1.4 Organization

This thesis consists of six chapters. In the first chapter, after the introduction section, we included a literature review on the identity system. Then, we mentioned about the contribution of this thesis.

Chapter 2 will give the necessary definitions about Cryptographic Primitives, Verifiable Credentials and Distributed Ledger Technologies. After describing underlying technologies, we defined Self Sovereign Identity and its multilayered structure. Lastly, the working mechanism of credential management is described.

Chapter 3 is focused on Indirect SSI controls. After we briefly explain what Indirect SSI control is; Guardianship, Guardianship life cycle, risk and challenges are described. Also, this chapter includes conceptual guardianship model and proof of concept use case.

Chapter 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 **Chapter 5**, we analysed the proposed framework. Comparative analysis of our framework and existing system was made.

Chapter 6 summarizes and concludes the manuscript.

Chapter 2

Preliminaries

In this chapter, we are going to summarize the necessary notions and definitions that will be used in the following chapters. As it is already stated earlier, even if there are different SSI proposals, we are going follow the definitions and notions given by W3C Verifiable Credential reference documents and Sovrin Foundation, please refer to [2] and [5] for further details.

2.1 Cryptographic Primitives

For further details of cryptographic primitives, please see [55] and the references therein.

2.1.1 Digital Signatures

Digital signatures are the public-key counterpart of message authentication codes, and their syntax and security guarantees are analogous [55]. Signature schemes allow a signer who has private-public key pair, to "sign" a message using the associated private key in such a way that anyone who knows signer's public key can verify that the message originated from the signer and was not modified in transit. Digital Signatures are used for authentication of messages or documents. Industries that utilize digital signatures can be extensively listed as government, healthcare and finance. Many countries have laws stating that digital signatures are equivalent to wet signatures [56]. In [57]; Rivest,

Shamir and Aldeman proposed digital signature algorithm for using in Public Key Cryptosystem, also known as Public Key Infrastructure (PKI). Digital signatures rely on PKI. This means there must be two cryptographic key for the signing. There are several digital signature schemes such as DSA, RSA, El-Gamal, ECDSA. All of them rely on hash function under the skin.

2.1.2 Hash Functions

A hash function is a one-way function that produces fixed-size output. One way means input cannot be obtained from the output. Most popular hash algorithm can be listed as SHA-1, SHA-2, MD5.

If we examine the Digital Signature working mechanism in details; First of all, the data to be signed hashed by using hash algorithms. Thus, we have a fixed size digest data independent of the size of the original data. Then PKI comes into play. This digest data is signed from the key pair we have, using the private key. Thus, the signed document is obtained. A person who wants to verify this signature must use the public key of the signer. By opening the signed document with the public key of the signer, he/she gets the digest of the document. Then, it is ensured that the message does not change by comparing the hash of the document with the digest obtained.

2.1.3 Camenish Lyanskaya (CL) Signatures

Camenish Lyanskaya (CL) signatures is a digital signature scheme; CL signatures play an important role in anonymous credentials. It was introduced in 2002 by Jan Camenisch and Anna Lysyanskaya [58]. Its most important feature is that it gives more importance to privacy and anonymity. In digital signatures, the public key of the person is known, and thanks to this, the signature can be verified. However, this situation reveals the identity of the signer. Because public keys are known to whom. So what if he/she does not want to reveal his / her identity? As a solution to this problem, Anonymous Credential Systems was introduced. The most used is CL signature. Public keys are not known in CL signature. CL signature is essential because Sovrin Network, which we used in this study also relies on CL signature. We have examined Anonymous Credentials in detail in Section 2.2.2.

2.1.4 Zero Knowledge Proof

A zero-knowledge proof (ZKP) makes it possible to prove a statement is true while preserving the confidentiality of secret information [59]. This makes sense when the veracity of the statement is not obvious on its own, but the prover knows relevant secret information (or has a skill, like super-computation ability) that enables producing proof. The notion of secrecy is used here in the sense of prohibited leakage, but a ZKP makes sense even if the 'secret' (or any portion of it) is known a priori by the verifier(s).

In SSI schemes, ZKP protocols are used for anonymous credentials presentation, especially for fully or partially hidden credential presentation.

2.2 Verifiable Credentials

Credentials are used for attestation of competence via some physical document such as driver license, passport or university degree in the real world. These credential issued by trusted authorities. Holder of the credential must present this credential physically to verify to the third parties. In the digital world, credentials are similar to real life, but also they are cryptographically constructed. With these feature verifying operation involves some cryptographic calculations. But this makes credentials harder to forge. So in principle, Verifiable Credentials are cryptographically secured digital data that person can use to prove something.
World Wide Web Consortium has a working group for VCs, and they published the Verifiable Credentials Data Model 1.0 specification [2] in 2019 to form a common basis for all studies. In this study, we are following their notions. Before describing the VC Data Model, it is necessary to explain some concepts for a better understanding. There are some roles, actors and relationships in the VC Data Model shown in Figure 2.1

- **Holder:** An entity that possesses one or more VCs. Students or employees can be given as examples.
- **Issuer:** An entity that creates the VCs from claims. Corporations, organizations, the government can be given as examples. Also, individuals can be in the role of the issuer.

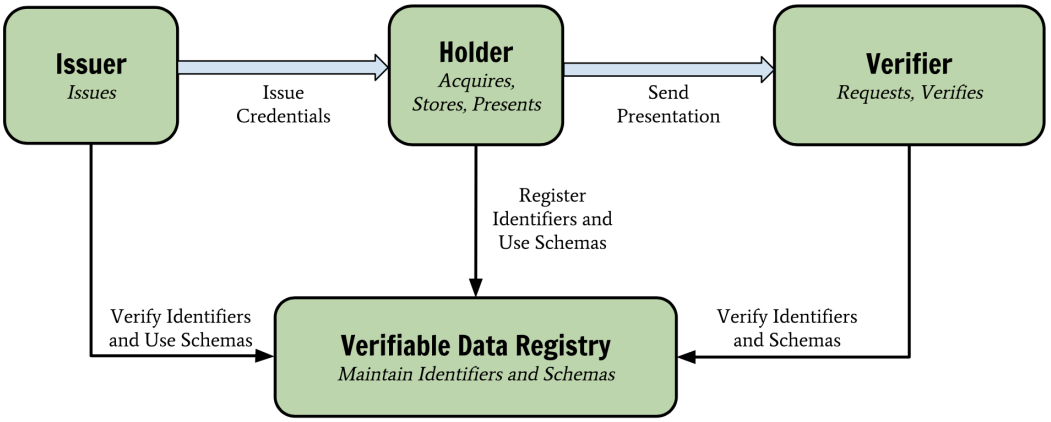


FIGURE 2.1: The roles and information flows in Verifiable Credentials [2]

- **Subject:** An entity that indicates whom VCs are given. Such as human beings, animals or things. In most use case subject is a holder itself. But in specific cases holder does not have to be subject such as parent hold the credentials that subject is their child. This is an example of Subject-Holder relationship and examined in details in a further chapter.
- **Verifier:** An entity that receiving the VCs for processing.
- **Verifiable Data Registry** is a distributed database that holds the VC schemas, revocation data, public keys etc.

As we briefly explained actors and roles in VCs, now we can examine more deeply the core data model. There are three core concepts in the VCs data model given below.

1. **Claims** are the statement that given for the subject of the credential. For instance, Alice has graduated from Faber College. Thus Faber College claims that Alice is an alumnus of Faber College.
2. **Credentials** consist of multiple claims. It also has a credential metadata such as issuer name, the expiry date of the credential, public key and so on. Usually, this metadata signed by the issuer of the credential. So this signature also takes part in the credentials as proof.
3. **Presentations** are the key concept of the VCs data model. Thanks to presentations concept, entities perform selective disclosure to prove their claims. Verifiable

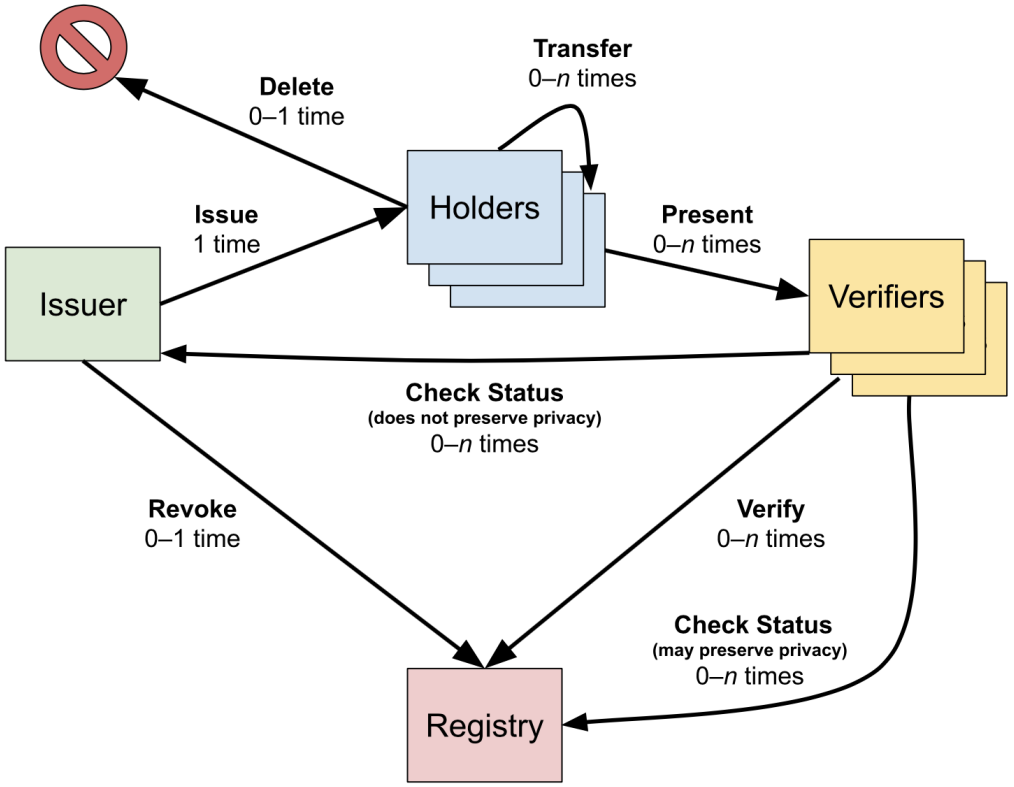


FIGURE 2.2: Life of Single Verifiable Credential [2]

Presentation (VP) is a subset of one's persona. Such as business persona or family persona. It might consist of one or more VCs.

In the VCs, there are much more concepts than the core concepts given above. These are *contexts, identifiers, types, credential subject, issuer, issuance date, proofs, expiration* and *status*. For Identifiers concept, please refer to Section 2.2.1.
Lastly, we have to examine the lifecycle of the verifiable credentials shown in Figure 2.2. We have follow, following notions: Issuer (I), Verifier (V), Holder (H)

1. I creates VC for H
2. H might transfer its VCs to another H
3. H present one or more VC in the VP to a V
4. V verifies the VP and VC.
5. I might revoke the VC
6. H might delete the VC

These six steps given above not fixed. Some steps might not occur.

2.2.1 Decentralized Identifiers (DIDs)

Identifiers are one of the important concepts in Verifiable Credential. Subjects have distinguished via identifiers. Giving credentials on behalf of something or someone based on the same specification is necessary. Thus W3C published specification about identifiers that will widely use in distributed systems called Decentralized Identifiers (DIDs) [32]. DIDs are not compulsory for VCs. Even so, it's expected that many software libraries will implement this feature for VC.
Globally unique identifiers notion is not new. In 1980 A Universally Unique IDentifier (UUID) was developed and later became standard in [60]. DIDs are essentially a new type of verifiable, decentralized, globally unique digital identifiers. They are similar to UUID except they have cryptographical material inside. In the centralized system, there are always authorities on the root, such as certificates authorities. These authorities serve as a root of trust on their own. With the rise of distributed ledger technology, entities such as people, organizations or things are not limited to use this centralized root of trust. Each entity can create identifiers for themself. With the DIDs, entities can authenticate using proofs such as a digital signature. DIDs have a corresponding DID document which includes several service endpoints and necessary information about DID.

Before diving into DID syntax firstly we will describe DID document core properties in detail.

- **Identifiers** are generally URLs or URIs that identify the subject.
- **DID Subject** is the entity itself whom DID document about.
- **Public Keys** are used for authentication and authorization with the related parties. They are used whenever digital signatures need.
- **Authentication** is a property that allows entity proofs it is the DID subject.
- **Authorization and Delegation** indicates that how the operation will be perform in the name of DID subject

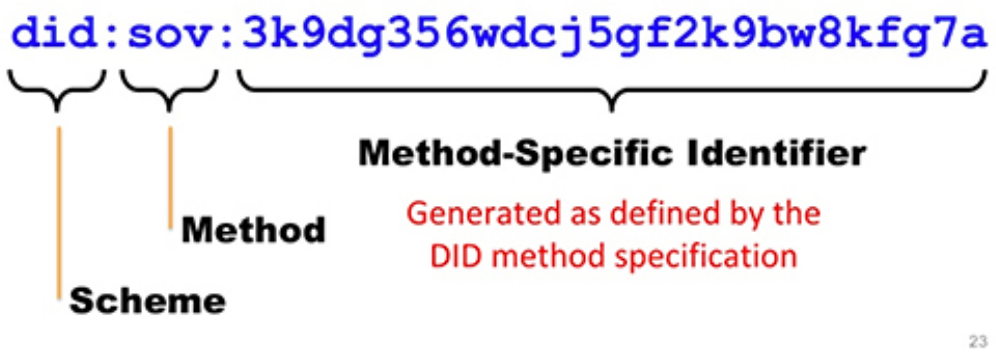


FIGURE 2.3: Syntax of Decentralized Identifiers [3]

- **Service Endpoints** are expressed the way of communication with the DID Subject.
- **Created** is a xml date time value that DID document created.
- **Updated** is a xml date time value that DID document updated.
- **Proof** is a cryptographic proof of the integrity of the DID document

DID syntax relies on URI standards [61]. A simple DID example is a string consist of three parts as follows as show in Figure 2.3 with refer to [32]:

- URL scheme identifier (did)
- Identifier for the DID method
- DID method-specific identifier.

The scheme specifies how a DID was constructed. Since DIDs are a URI format, the DID is written in this format. For example, the URL is another URI format. The DID Method specifies which network or distributed ledger the DID scheme is implemented by. DID method-specific identifier is a document covering all aspects of a DID method.

DIDs have two important advantages over traditional PKI systems [62]. The first is to make identity management easy on behalf of someone else. This is a must-have feature for indirect identity management systems such as guardianship and delegation. The other advantage is that it enables easy and straightforward communication with the DID subject. In Table 2.1, comparison of DID with other systems is given over specific properties.

Table 2.1: Comparison of DIDs with URLs and Email Addresses [12]

Property	**URL**	**Email**	**DID**
Persistent	No	No	Yes
Resolvable	Yes	No	Yes
Cryptographically verifiable	No	No	Yes
Decentralized	No	No	Yes
Human-friendly	Yes	Yes	No

2.2.2 Anonymous Credentials

First introduced in [63] by Chaum. Then it fully realized by Camenisch and Lysyanskaya in [64] and later developed the following works [65], [66], [51]. It allows the users to prove their credential without disclose other than necessary information. Underlying technologies are CL signature and Zero Knowledge proof mentioned in Section 2.1. Anonymous Credential has significant properties such as unlinkability. This means that it is impossible to correlate between the credential and the issuer or verifier. This is completed via Zero knowledge proof. Holder presents the zero knowledge proof rather than an actual credential. Anonymous credentials are also unforgeable thanks to the strong cryptographic foundation. They have preferred mostly privacy-preserving application.

Protocol Description

We have followed following notions in describing protocol.

- L is the attributes number in the credential and represented as $m_1, \ldots, m_L$

Credential has a three type of attributes explained in [67]

- **Issuer attributes** represent as $(m_i)_j$. Include the meta data about credential and issuer
- **Subject attributes** represent as $(m_s)_k$ Include the claims that the issuer made.
- **Blind attributes** represent as $(m_b)_l$. Only known by the holder.

Workflow for creating anonymous credential looks like below.

1. Create the credential schema that has a list of attributes m_s for subject. This attributes can be listed as follows

- Type of digital signature.
- Indices of hidden attributes
- Public key P_k
- Revocation info

2. Holder receive the credential schema and add the hidden attribute that not defined in credential schema. For this Holder generates 256 bit random number K. Then sends Credential request to the Issuer

3. Issuer receive credential request and generates credential pair C_p, C_{NR} for Holder. C_p, is a requested credential. C_{NR}, is a non-revocation info of the credential.

4. Verifier sends the Proof Request P_R to the Holder. The P_R includes
   - Credential schema
   - Disclosure predicates

5. Holder firstly check the credential schema and verifies it. Control the non-revoke info. Later, creates a Proof P and send to the Verifier.

6. Verifier verifies the P and accepts the credential is valid.

Proof presentation is important subject. Holder can create proof statements for verifier. These statements can be hidden or revealed. In the revealed statements attribute value is fully exposed. In the hidden, there are three level:

1. **Fully hidden:** No attribute value is exposed. Only information that disclosed is the Holder knows the value.

2. **Linked:** Holder disclose the information that two attributes have the same value but not the value itself.

3. **Partially Disclosed:** Holder disclose the some predicate about attribute value. Predicate consist of $<, >, \leq, \geq, \neq$

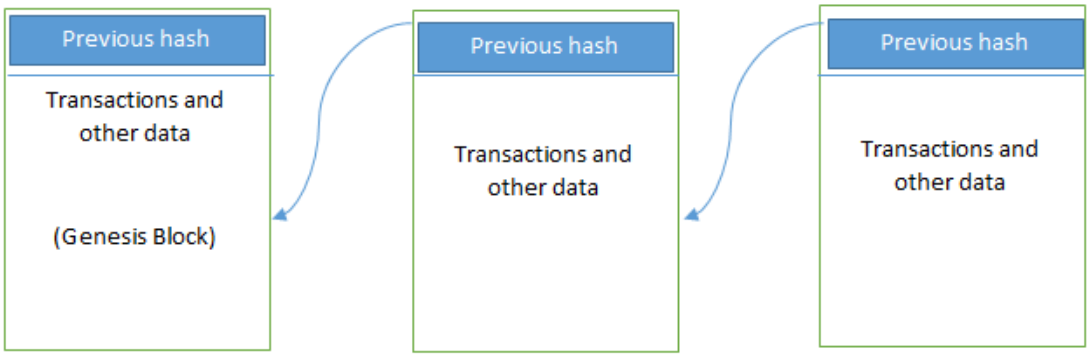


FIGURE 2.4: Generic structure of a blockchain [4]

2.3 Distributed Ledger Technologies

Distributed Ledger Technology is a data storage without the central authority. It relies on Blockchain technology. To understand the DLT, we must first examine Blockchain. In recent years with the rise of the cryptocurrencies like the Bitcoin, blockchain becomes much popular. It starts with the paper *"Bitcoin: A Peer-To-Peer Electronic Cash System"* by Satashi Nakamota in 2008 [34]. This paper is the first realization of the truly decentralized system. It allows the user to send the electronic currency without need any 3rd party authority. Since that day popularity of this technology has never ceased to increase.

Blockchain is a cryptographic, irreversible data structure that contains transaction blocks added one after another. It runs on a network with multiple users or nodes. When prompted to add a new record, firstly it must be verified by the nodes on the network. Then it is added to the end of the chain in a new block. The chain is append-only. Old blocks cannot be altered in any way as new blocks are added. To achieve this, it uses the structure shown in Figure 2.4.

Each block contains the hash of the previous block. These hashes are digitally signed. If a block is changed, then its signature will not be verified because its hash will change. In this way, the integrity of the chain is preserved.

Blockchain can be public or private. In the public blockchain, everyone can read from the chain. But adding a new record can be allowed or not allowed to everyone. This is called *Permissionless* or *Permissioned* blockchain. For instance, Bitcoin uses a public permissionless blockchain. Because anyone who does solve some cryptographic problems can add a new record on the bitcoin. In the other hand, Sovrin Network uses public permissioned blockchain. This means anyone can read from the chain but adding new

record is allowed only for designated nodes.
Blockchain has no trusted third party for the validation of transactions. Instead of it uses a cryptographic proof to validate. Every transaction is digitally signed and broadcast to the network. Thus any node can verify the signature.
Blockchain technology may have started in the finance industry, but recently it has new applications beyond finance. For instance, government, IoT, health etc.

DLT applications are not limited to finance. Such as digital identity, trade/commerce, Governance, Healthcare application is on the rise in recent years. Sovrin foundation published an article on the use of DLT in the humanitarian sector in 2019 [68]. In this article, comparative analysis of DLT and traditional system solutions were made through use cases such as cash transfer, identification for refugees and health service.

Advantages of DLT can be listed as below as stated in [35].

- **Decentralization and disintermediation:** DLT allows the transfer of tokens or digital assets between the two parties without the need for an intermediary institution.
- **Greater transparency and easier auditability:** All members in the network keep a copy of the Distributed Ledger. A change is taken by consensus decision and broadcast to the entire network. In this way, frauds are prevented without a central authority.
- **Automation and programmability:** Thanks to technologies such as smart contract in DLT, it can automate many transactions. For example, a system can be designed to pay automatically when certain conditions are met.
- **Immutability and verifiability:** Thanks to the blockchain technology used on the basis, it is ensured that the transactions made cannot be changed. It also allows anyone to see and approve transactions registered on the blockchain.
- **Gains in speed and efficiency:** It provides the acceleration of the process by removing the intermediary institutions from performing the transactions.

- **Cost reductions:** The increase in performance also brings with it a reduction in costs. In addition, less fraud decreases costs. According to [69], the financial industry can save 15-20 billion dollars a year with DLT.
- **Enhanced cybersecurity resilience:** It offers more security than traditional centralized database systems. For example, centralized systems are open to a single point of failure by nature.

2.4 Self Sovereign Identity

SSI is the most advanced point of identity management systems. It is a truly decentralized system using the components described in Section 2.1, 2.2, 2.3 as an underlying technology. Although many different companies are working on SSI, Sovrin Foundation was taken as reference in this study. The study conducted on the Sovrin Network. Therefore, the technical infrastructure of SSI will be explained over the Sovrin Network.
Sovrin Foundation set out to solve the identity problem of the Internet in 2016 [11]. It is designed to be a new Internet identity layer in a decentralized structure with a public permissioned blockchain.
Sovrin Framework has a 3-layer structure [5]:

- **Sovrin Ledger** is the most fundamental component. It is a distributed ledger maintained by trusted institutions.
- **Sovrin Agents** is the layer that provides service for identity management operations.
- **Sovrin Clients** applications that end-users use on their devices. It provides communication with Sovrin agent and ledger.

2.4.1 Sovrin Ledger

The consensus protocol used in the blockchain in this layer is Plenum Consensus Protocol. The Plenum is the enhanced form of Redundant Byzantine Fault Tolerant (RFBT) protocol [70]. Because it has a public permissioned structure, it is governed by institutes

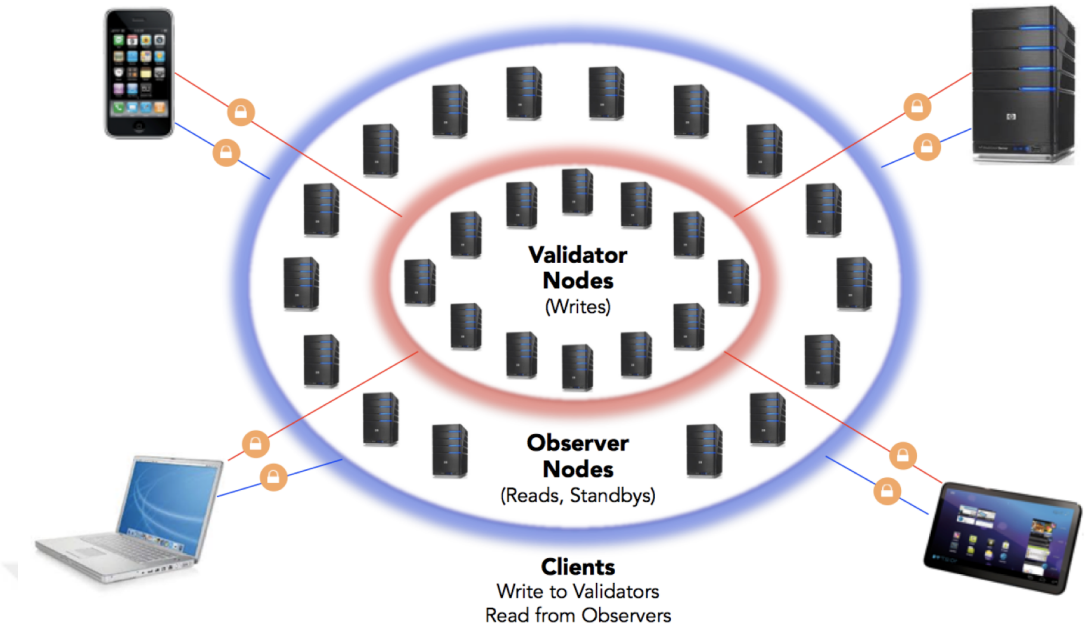


FIGURE 2.5: The two basic node types in the Sovrin ledger [5]

called Board of Trustees. This board consists of Stewards. Stewards can be any institution that wants to be involved in the SSI system, such as universities, hospitals, telecom companies. Nodes in the ledger are operated by stewards. There are two types of nodes, as shown in Figure 2.5.

- **Validator nodes** are allowed to both write and read a new transaction into the ledger. They run the Plenum protocol to validation.
- **Observer nodes** are only allowed reading from the ledger.

There are four different ledgers in Sovrin Network for different purposes.

1. **Identity Ledger** is the main ledger in which the identity information of the users with Sovrin identity is kept.
2. **Pool Ledger** is the ledger where node information is kept. For instance, which node has the authorization to write.
3. **Voting Ledger** is the ledger where the votes given by the stewards are kept.
4. **Config Ledger** is the ledger where the settings that concern the entire network are kept. For example, info like at least how many stewards must be approved for a new steward to join the network.

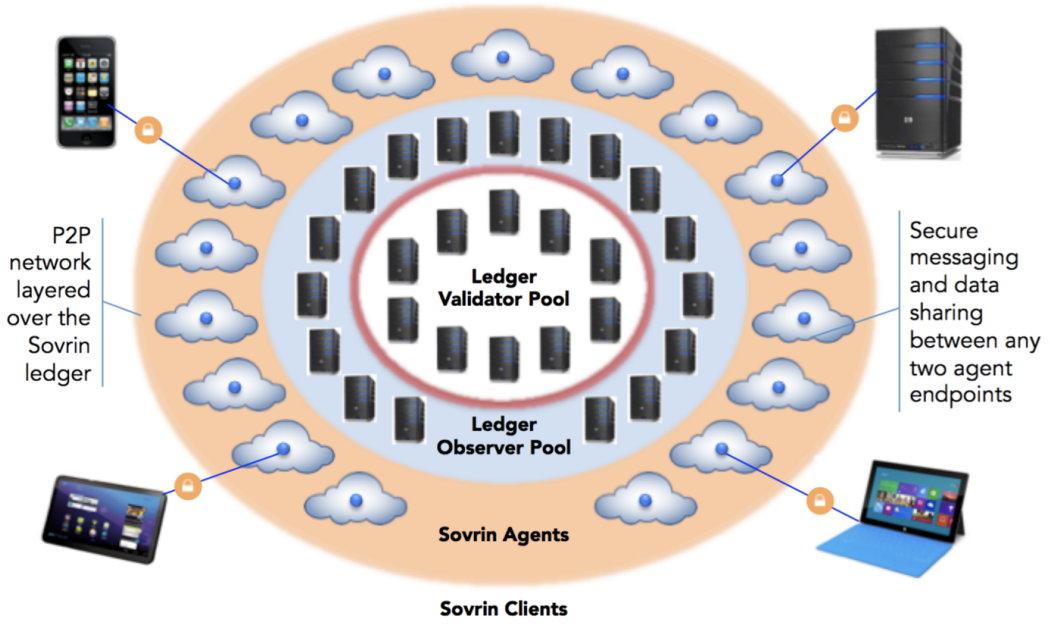


FIGURE 2.6: Sovrin agents of the Sovrin network [5]

Validator nodes and Sovrin clients verify each other when they communicate. They do this through their digital signatures. Thus, with mutual authentication, they ensure that they are communicating with real nodes or clients.

2.4.2 Sovrin Agents

Sovrin agents are the middle layer of the Sovrin Framework, as shown in Figure 2.6. Sovrin agents are not obligatory for the system. Clients can talk directly to the ledger. But agents make the system easier to use for end-users. Agents perform the tasks listed below [5].

- **Persistent P2P messaging endpoints:** Sovrin clients generally run on smart devices such as mobile phones. Usually, they do not have their endpoints for messaging. Agents provide this endpoint service to clients. Thus, they can be found by the network.
- **Coordination endpoints for multiple clients:** Agents undertake the task of transmitting messages between clients.

- **Encrypted backup of Sovrin keyrings:** Agents also take care of the clients key. They facilitate situations such as key recovery by keeping their encrypted backups of client's key in their systems.
- **Encrypted data storage and sharing:** Agents give the opportunity to store their data encrypted for clients.

One of the ten key principles of SSI is portability. In Sovrin, the data of the identity holders are fully portable. It does not matter which agent they use because the information is kept in a general independent ledger. Identity owner has full control over agent endpoints. They can switch to a new agent as they wish or stop using it completely. In Sovrin, portability is as important as security and privacy.

2.4.3 Sovrin Clients

Sovrin clients are the last layer of the Sovrin framework. It is the most critical layer since it contains the management of the private keys of the users. Because the private keys of the identity holders are the most important asset in the system. The first task of client applications is the management of the user's keys. A second task is to store the user's information locally securely.
When client applications are wanted to be used for the first time, they must first connect to Sovrin network. If the user does not have a previous Sovrin Identity, then it should be connected to the Sovrin Identity holders called Trust anchor. Trust anchors have the right to add new identities to the network. We can summarize this process as follows.

1. First of all, the identity owner connects to a trust anchor through his/her client to prove his/her identity.
2. Trust anchor returns the challenge token in response.
3. Sovrin client creates a key pair and adds it to the key chain of the identity holder.
4. Sovrin client signs the challenge token with the private key it creates.
5. Trust anchor verifies the signature. In this way, It authenticates the new key pair. then It registers new Sovrin identity to ledger.
6. Trust anchor signs the transaction result and sends it to the client.

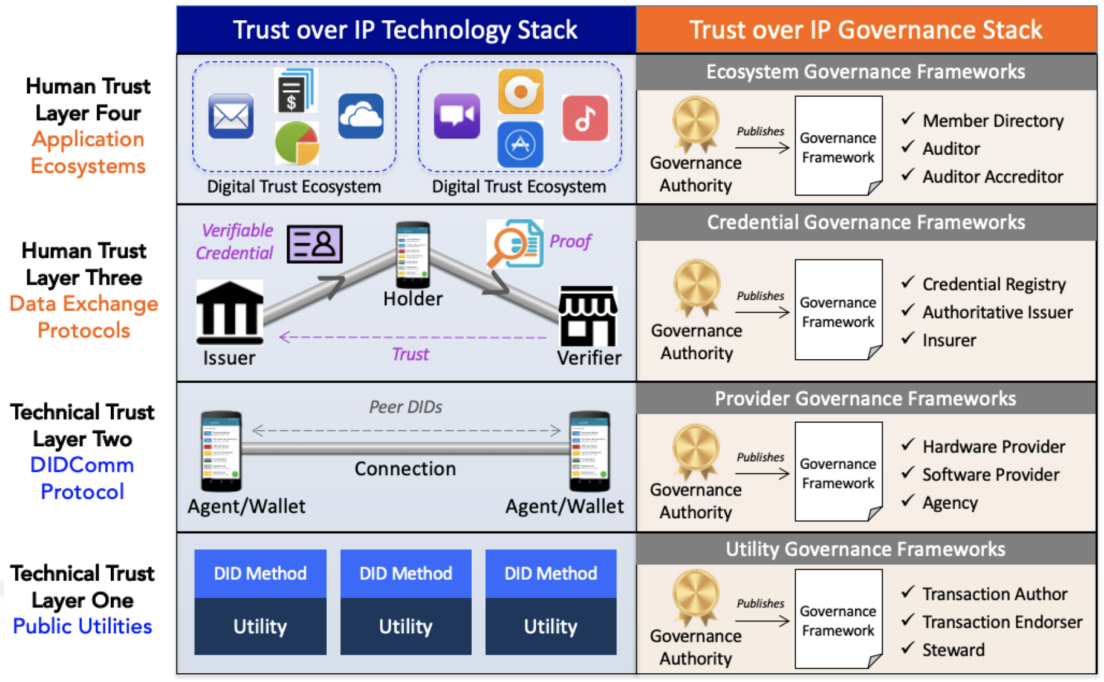


FIGURE 2.7: The four layers of the Trust over IP stack [6]

7. Client verifies the signature. Identity owner is now registered to ledger. He/she can start using his/her agent.

One of the favourable aspects of SSI is that identity holders to use their identities on any application, device or network. For this, it is sufficient to have Sovrin client installed at the desired location. Once registered with ledger with the trust anchor, subsequent client uses will be much simpler. Because it will be able to directly authenticate itself between the client and new agent without the need for trust anchor.

2.4.4 Sovrin Infrastructure

Sovrin uses the technologies described in the previous sections in its infrastructure. However, the layers using these technologies differ. Let us examine this situation closely.

Sovrin architecture consists of 4 layers. This architecture is called the Trust over IP (ToIP) Stack. It is defined in [6]. ToIP is designed to support digital guardianship fully. 4 layers can be seen in Figure 2.7. In this section, we will examine these four layers of SSI architecture in terms of guardianship.

- **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 [37] 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 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 via DIDComm protocol. It is the layer where the basic guardianship tasks are performed. Because 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. Figure 2.9 shows a wallet distinction in the guardianship relationship between a mother and child.

- **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 take place 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 layer one to verify Issuer's DIDs.

  In this layer, Sovrin uses the Hyperledger Aries library [38]. Hyperledger Aries is an open source library created to perform agent tasks mentioned in Section 2.4.2. Using Aries, people can develop agent applications for SSI. Aries library is a platform-independent library, as seen in Figure 2.8. 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

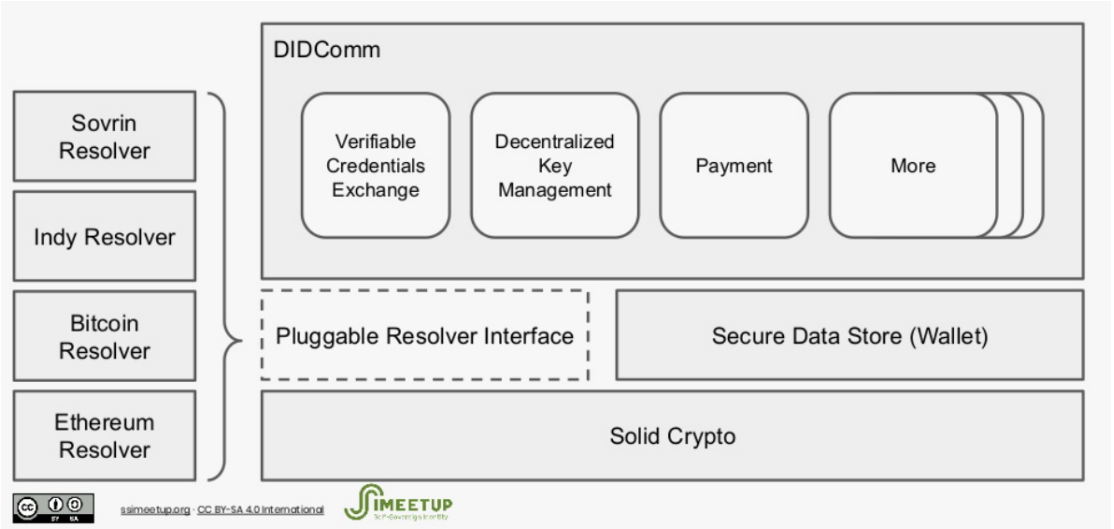


FIGURE 2.8: Hyperledger Aries Architecture [7]

a ledger in its infrastructure, is used. The Indy library acts as a resolver for DIDs, schemas and credential definitions.

- **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 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 properly. A governance framework structure that is special to guardianship has been created within Hyperledger Aries in 2019 [71]. Aries agent framework has libraries created for different software languages such as Python, .NET, GO. In this study, Aries Agent Python library [72] 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 most important tasks of the agents is verifiable information exchange. Let us take a closer look at how this workflow is in the next Section.

2.4.5 Sovrin SSI Credential Management

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

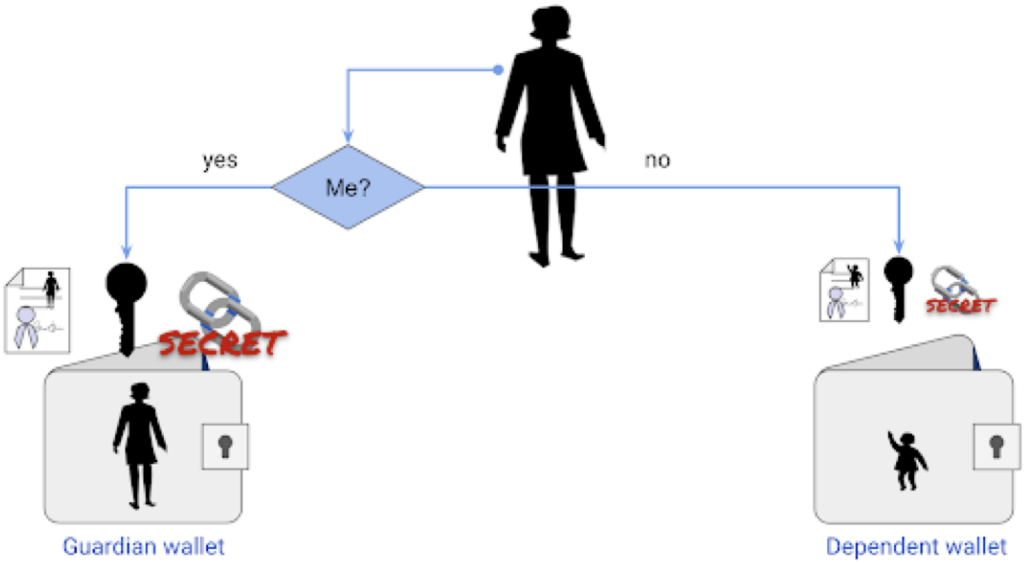


FIGURE 2.9: Digital wallets for guardian and dependent [8]

- An issuer issues a verifiable credential to a holder. Issuance always occurs before any other actions 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 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 2.2 demonstrates life cycle of a single verifiable credential. Figure 2.10 demonstrates workflow entities in SSI architecture.

Self sovereign identity relies basically on attributes. An attribute explains in [73] as "what qualifies a person, without necessarily being unique to that person." These are elements

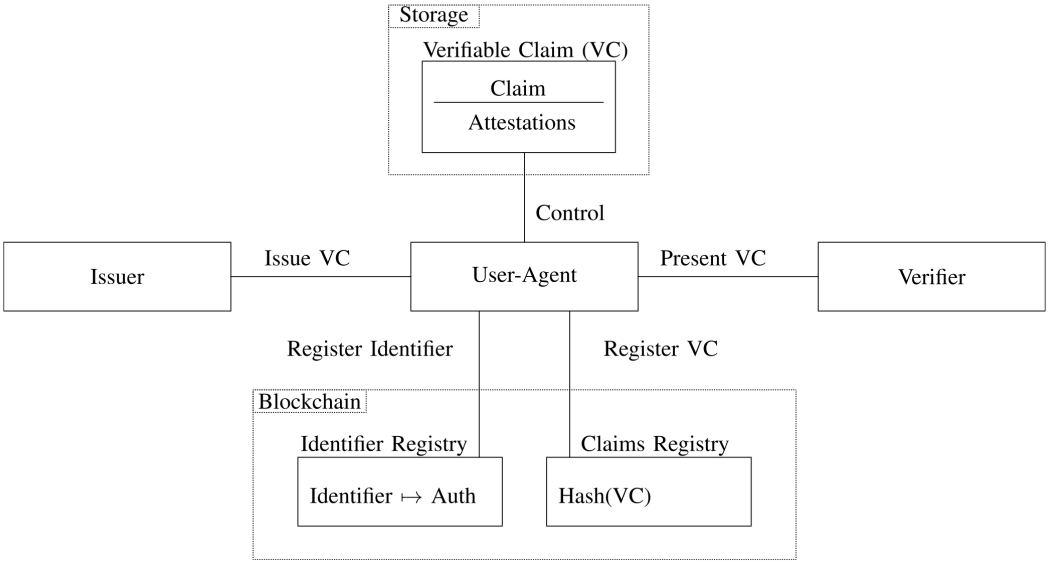


FIGURE 2.10: Self Sovereign Identity Architecture [1].

such as gender, weight, height etc. The attribute can also be verifiable credentials. The verifiable credential has three entity. These are

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

Simple workflow between these entities explained below with reference to [67].
To simplify, the example has one credential and single I, H and V entities. But in the real life, Entities can be multiple and holder can have multiple credentials.

- First of all, 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.
- 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.
- Thereafter holder sends a credential offers to Issuer to get a credential.

- Issuer receives this offer, fills the attributes that are written in the schema and publish 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.

Chapter 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 should 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.

Considering the entire life of the person from birth to death, any scenario should be operable in these systems. For instance, the management of 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.
There are many situations where a person cannot manage their own credentials. To illustrate:

- **Older people:** Elderly people with limited capacity to use technology may need someone else's help for credential management.
- **Sickness:** Examples include people who have physically limited capacity or disease that prevents them from managing their own credentials, such as dementia.
- **Children:** People under the age of 18 who will not make their own decisions legally.

- **Child refugee:** In particular came up with the growing refugee crisis in recent years, the situation of unaccompanied children in refugee camps.
- **Renting:** In the case of renting a car or house, management rights for that item are temporarily transferred to someone else.
- **Verdict:** In various cases, courts can give custody of one person to another.
- **Power of Attorney:** We can give delegation for certain situations. For example, a mother can give a delegation to the teacher at school for her child. Or an elderly person can give their child a power of attorney to take their medication.
- **Pet Ownership:** If we consider that animals have an identity and digital credentials, they will also need to be controlled by their owners.
- **Smart Device Ownership:** With the smart devices rising in recent years, how to manage the digital data of these devices has come to the agenda.

For those who cannot directly access or manage their own identity, a mechanism is required that will provide the following conditions [8]:

- Technically able to work with existing identity systems
- Support existing legal and social processes
- Revocation support

Guardianship is the mechanism that provides these conditions. The Guardianship subject has been an important component of SSI from the very beginning. For this purpose, Sovrin Guardianship Working Group, established under the Sovrin Foundation, aims to provide a basis for future studies on this subject. Here, the Guardianship Whitepaper [8] published by SGWG was taken as reference.

The Sovrin framework defines three different roles as indirect identity control relationship described in [8].

- **Guardianship** is the type of relationship that occurs when one of the two sides cannot control their digital credentials. The critical point here is that usually,

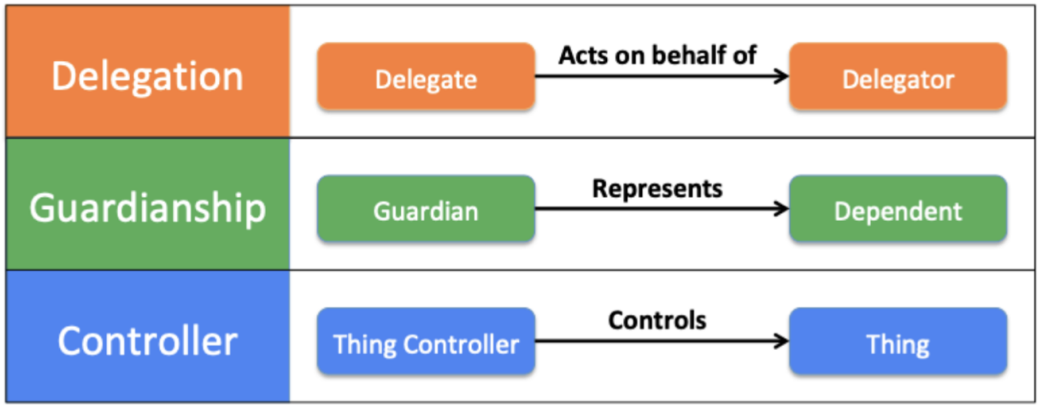


FIGURE 3.1: The three types of indirect identity control relationships [8]

both sides are able to control their digital ID records. However, due to various situations, one side may not be able to control the identity records. Elderly people with dementia, children, or children in refugee camps can be given as an example of this type.

- **Delegation** is one of the most used control relationship types. Delegates entity act on behalf of Delegator's entity. The main difference between Guardianship and Delegation is that both the delegator and delegate are both capable of managing their digital assets in the delegation. So they both have control of their private key. Delegator delegates some responsibilities by publishing delegation credentials. Renting and Power of Attorney can be given as examples for delegation.

- **Controllership** can be defined as the management of the digital assets of an object that can never be self-sovereign. With the concept of the Internet of Things in our lives, it is inevitable for our smart devices to have a digital identity. The Thing can be any smart device. The person who controls the digital identity of this device is called Controller. Controllership is not just a type of relationship built on smart devices. Pets and their digital identities can also be viewed as controllership relationship. The controller can publish controller credentials for others so that the person who has the credential can perform the authorizations written in the credential. The best examples of Controllership are pet ownership and smart device ownership.

These 3 control relationships can be depicted as in Figure 3.1. These indirect relationship

types are not completely independent of each other. For example, most guardianship relationships include delegation. Likewise, the guardian often takes some authority and responsibility with his controller credential. As a result, these three relationships often work together. Table 3.1 summarized the key distinctions of the these relationships.

Table 3.1: Primary distinctions between the three types of identity control relationships [13]

	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

As stated earlier, SSI guardianship is the main focus of this thesis. Therefore, from now on, we are going to dive into indirect SSI control within the guardianship context. However, as noted above these notions would also apply for the other settings.

3.1 Guardianship

The need to cover everyone is best explaining why there is a need for guardianship in SSI systems. 3.3 billion people in the world live without access to Internet [74]. This number includes 1.9 billion children [8]. This shows that when designing SSI systems, we need to reconsider the functioning of relations between identity holders and other processes.

Considering the real life restrictions, Sovrin Foundation is looking for a solution to this problem without giving up the basic principles of SSI. For this purpose, three indirect identity control models are defined and summarized in Table 3.1. In this section, guardianship, one of these three models, will be examined in detail.

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 also guardian relationships formed by legal obligation. For example, power of attorney or conservatorship. In both cases, both formally and informally, thanks to the guardianship, someone in need are protected and helped. In SSI Systems, Guardianship involves fully controlling the dependent's digital wallet. This includes sharing proof when necessary. Naturally, all SSI mechanisms are also valid in guardianship.

There are different scenarios where guardianship is needed. As an example, when people do not have access to 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 given as a real-life example. 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 by 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 three.

1. **Full guardianship:** It is a situation where 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 [75, Section 2.10.7] as follows:

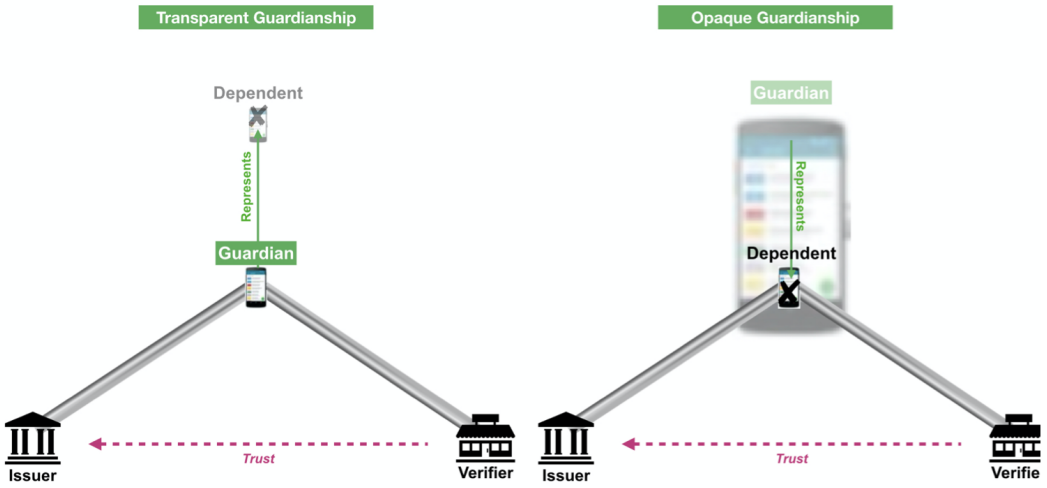


FIGURE 3.2: Transparent vs. opaque guardianship [8]

> "**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. A 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 3.2 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 the existence of the guardian. 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 dependent's social life.

3.2 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 the operation of existing systems 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 actually means issuing certain digital credentials for both the guardian and the dependent. But 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 3.3 and can be summarized as follows.

1. **Inception:** The first step is to determine the need for the 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 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 described in Section 2.2. 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 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.

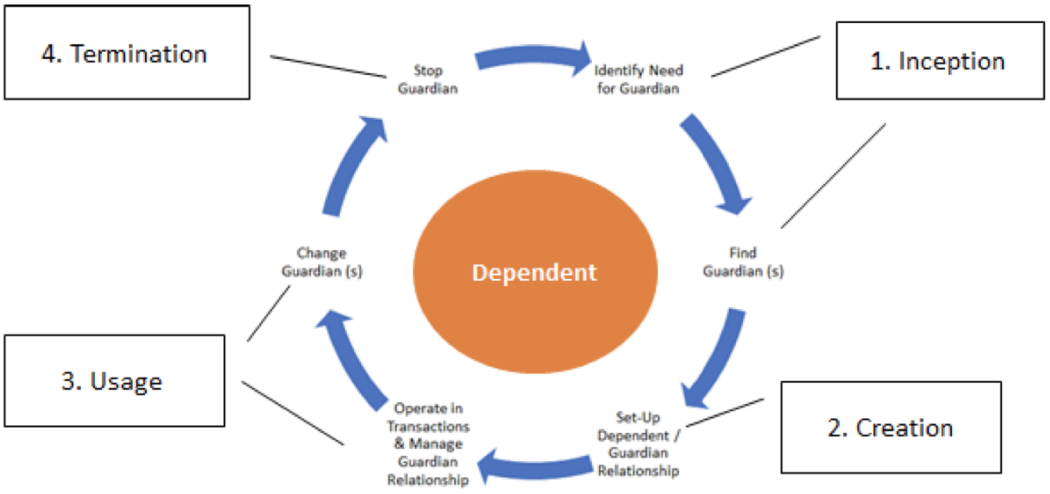


FIGURE 3.3: The four major stages in the guardianship life cycle [8]

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 a revocation of the credentials given for guardianship. In this way, the authority of the guardian to access cryptographic keys is eliminated.

3.3 Indirect SSI Control Risks and Challenges

3.3.1 Risks arise within Guardianship

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

- **Inherent Risks**: SSI is the self-management of the person's own identity data. But when the guardian relationship is thought, it seems to contradict this explanation. Guardianship is a mechanism that gives control of one's identity to someone

else. Due to the nature of this mechanism, certain risks occur. The first is that Dependent becomes too dependent. Typically, guardianship is a temporary situation. The relationship ends when Dependent is able to manage their digital identity. However, this is not always the case. For example, elderly people with dementia can remain in this condition until the end of their life. However, the guardian should not continue his/her duty when the dependent is able to take control. For this, the guardian should be prevented from making any financial gain from this relationship. The second is the re-centralization. SSI systems are decentralized and exempt from a single point of failure. But some institutions and organizations are at risk of using the guardianship mechanism to control their customers.
Inherent risks are very general and basic 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, the guardian acts in the best interest for the dependent. But this concept of best interest is subjective. A situation that would be in the interest of dependent could be an unwanted situation by the guardian. In such scenarios, the path to be followed, and the best interest concept should be clearly defined. To go through the example scenario, the dependent person does not want to use the drug that doctors recommend to use. What kind of attitude should the guardian take in this case? The opposite may also be valid, a situation that the dependent wants is rejected by the guardian.
  As a possible solution to this risk, a certification requirement may be introduced to become a guardian. At the same time, 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 a dependent.
  - Dependent's wallet should always be auditable.
  - During risky transactions, the 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.
  In order to prevent risks arising from such situations, frameworks should be designed to work with each other. At the same time, high-quality user experience should be provided for users. Thus, it can be managed more easily.

- **Risks At Moments Of Transition Or Change**: Guardianship has a life cycle. Therefore, more than one guardianship can start in a person's life. These transitions should be done as defined in the guardianship life cycle. There should be no unterminated guardian relationship, although it is over. To prevent unwanted situations, it is necessary to use a framework that covers all processes from end to end. These processes can take place online or offline. When switching from an offline environment to an online environment, synchronization with the network must be done.

3.3.2 Challenges

In a guardianship, there are challenging element. There are technical difficulties to represent the social context. Let us dive in these challenges.

- **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 elderly person 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. However, 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 dependent's medical needs, but the dependent himself is responsible for the management of 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.

3.4 Guardianship Conceptual Model

Let us examine the elements required to create an SSI model in guardianship. SGWG specifies three building blocks to be used in guardianship modelling as follows [9]:

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:** Unique guardianship credential indicating the content and the boundaries of the relationship between the actors should be used.

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

Guardianship Trust Framework

Trust framework is a crucial component for guardianship in terms of its legal basis and

reliability. According to [9], a guardianship framework should contain the following necessity:

- A framework should be kept at an accessible address with the version number.
- It should be stated in which country it is valid and which legal restrictions it has.
- Guardianship justification should be stated. For instance `kinship` to indicate affinity relationships or `court_order`.
- It should be noted what the recommended behavior patterns for actors are.
- The authorization of the permissions granted in the guardianship must be specified. Such as `financial` or `medical`.
- It should be noted which restrictions exist.
- It should be stated which audit mechanisms are required or recommended.
- It should be stated which appeal mechanisms are required or recommended.
- It should be noted what are the best practices for guardianship challenges.
- It should be stated which freshness rules are used for offline use and revocation.

Guardianship Credential

Guardianship credentials are essentially verifiable credentials in the W3C VC standard. It can be in various data formats such as JSON-LD. According to [9], they should have certain characteristics listed below.

- ‘‘`https://www.w3.org/2018/credentials/v1`’’ must be added in the `@context` field. This applies to all VCs.
- The `type` field should be in the format shown in Figure 3.4. For the `form` field, `D` can be used for Delegation, `G` for Guardianship, and `C` for Controllership. The name of the utilized framework should be written in the `trust framework` section. `tfver` specifies the version of the framework used. `variant` shows the scheme used. For a complete example, `Proxy.G/UNICEF Vulnerable Populations Trust Framework /1.0/ChildGuardian` can be given with reference to [9].



FIGURE 3.4: Guardianship type field notation [9]

- Metadata fields must contain `trustFrameworkURI`, `auditURI` and `appealURI` information.
- The `credentialSubject` field can be divided into two as `holder` or `proxied`. `holder` represents delagate, guardian or controller. `proxied` represents the delegator, dependent or controlled thing.
- In the `credentialSubject.holder.type` field, the schema information for the `credentialSubject.holder` in URI format must be defined.
- In the `credentialSubject.proxied.type` field, the schema information for the `credentialSubject.proxied` in URI format must be defined.
- Credential may contain additional information under `credentialSubject.holder` field, such as the name of the holder, DID or biometric.
- Credential must contain additional information under `credentialSubject.proxied` field, such as the name of the proxied or biometric.

Guardianship Challenge

Guardianship challenge is the process where the holder proves that it has control over the identity of the dependent. The most important step in this process is the Verifiable Presentation (VP) request. Thus, the evaluation starts with this evidence. In this evaluation, the role of the holder and their permissions are compared. Restrictions are controlled. According to [9], the holder must reveal the information below in the VP.

```
1 @context
2 type
3 issuanceDate (if present in schema)
4 expirationDate (if present in schema)
5 credentialStatus (if present in schema)
6 issuer
7 trustFramework
8 auditURI
```

```
appealURI
credentialSubject.holder.type
credentialSubject.holder.role
credentialSubject.holder.rationaleURI
credentialSubject.holder.constraints.*
credentialSubject.proxied.type
credentialSubject.proxied.permissions
```

In addition, the holder must prove the authenticity of proxied. She can do this as specified in the trust framework she uses. It is usually done by providing extra information under the `credentialSubject.proxied` field.

In the view of above the information, to be self-contained, we are going to include how to create a guardian credential through a use case on the story of a girl in the refugee camp created by Sovrin Foundation in the next section.

3.5 Proof of Concept Use Case: Refugee Camp

Guardianship takes an important place among the principles and policies of SGF [75, Section 2.2 and 3.2]. For this reason, SGWG, which was created specifically for guardianship, studies this topic and explains the recommended mechanisms through use cases. SGWG examined the case of an elderly man with Dementia disease through a character named Jamie [76]. Another use case is a refugee girl named Mya [10]. Sovrin has studied this use case in detail and made examples using this story while guardianship is being discussed. We will explain the guardianship modelling described in Section 3.4 to ensure integrity, through the Mya use case, which can be summarized as follows.

Mya is 5 years old girl who has lost her family. She is in a refugee camp. Please refer to [10] for the detailed story. Mya's needs are summarized in Figure 3.5. Let us examine how a trust framework should be designed to cover these needs with reference to [71].

First, the guardianship rationale must be defined with following attributes.

- `dependent-appointment`: This is when dependent herself can appoint a guardian.

Identity Needs	
Simplicity	• Needs to prove identity without direct access to digital devices • Needs Guardian
Value	• Mya can be connected with family as and when they are found • Mya can access relief services to meet her basic needs (healthcare, food, shelter, security) • Mya can access education
Privacy	• Only extended family and trusted 3rd parties can claim relationship with Mya
Security	• Only extended family and trusted 3rd parties can claim a relationship with Mya or access her personal information

FIGURE 3.5: Mya User Story - Identity Needs [10]

- `kinship`: This is the situation when a guardian is assigned to a person who is dependent on age or disability. Guardian is related to dependent. So they can naturally claim guardianship status. Examples of kinship relationships are `biological-parent`, `step-parent`, `sibling`, `grandparent`, `aunt-or-uncle` and `first-cousin`.

- `adjudicated`: It is the situation where a guardian is determined by a legal authority.

- `self`: In the absence of a suitable guardian, a person is temporarily taking the guardian status of the dependent.

Then the attributes for the holder should be defined. But if ZKP is to be used, holder attributes should not be embedded in the guardian credential. It is sufficient to have a blinded link secret attribute as a holder attribute. For the dependent Identity, the attributes specified below can be included in credentials. Although iris and fingerprints are not mandatory, since Mya has no one, it is recommended to be used to identify her. The photo must be embedded.

```
1 credentialSubject.first_name
2 credentialSubject.last_name
3 credentialSubject.birth_date
```

```
credentialSubject.gender
credentialSubject.native_language
credentialSubject.identifying_marks
credentialSubject.photo
credentialSubject.iris
credentialSubject.fingerprint
```

The Guardian may have permission for the following items. These items must be defined in the trust framework.

- `routine-medical-care`: It includes normal medical treatments such as surgery, outpatient treatment, dental treatment, vaccination etc.
- `do-not-resuscitate`: Consent to discontinue life support. Such as permission to turn off dependent's life-support system.
- `school`: It includes permissions such as enrollment in the school, selection of courses.
- `necessaries`: It is the permit for receiving basic needs such as clothing, food and hygiene materials for dependent.
- `gender-identity`: Permission to determine the gender of the dependent.
- `religious-observance`: Permission to determine whether the dependent can perform religious duties.
- `light-travel`: Permission to take the dependent out until the sun sets.
- `extended-travel`: Permission to take the dependent out for a long time. For example, in the Mya use case, the guardian can take Mya out of the refugee camp to stay overnight.
- `unenroll`: Guardian may stop Mya from accessing camp facilities.
- `contracts`: Permission to make Legal or financial agreements on behalf of dependent.
- `marriage-family-planning`: Guardian can give consent for the dependent to get married.

- `delegate`: The guardian has the right to grant some permissions to a non-guardian.
- `successor`: The guardian has the permission to choose a replacement for itself.
- `authorize`: Authority to regulate permissions.

In order not to abuse these permissions in real life, the best practices below should be followed.

- When the guardian takes any action other than `school` and `necessaries` permissions, the guardian challenge should be performed.
- When using `routine-medical-care`, `do-not-resuscitate`, `gender-identity` and `light-travel` permissions, the photo of the dependent must be compared with dependent herself.
- Biometric matching should be preferred when performing `extended-travel`, `unenroll`, `contracts` and `marriage-family-planning` permissions.

As well as the permissions granted to the guardian, some restrictions were imposed. These restrictions in the guardianship credential must be defined in the trust framework.

- The first restriction is the restriction of the area where the guardian's powers apply. In this way, the guardian will be able to operate within specified boundaries. Following example shows how to define a boundary attribute.

```
"constraints.boundaries": "|en: West side of Euphrates
   river, within Baghdad city limits"
```

- Geographical restriction can also be defined by additional attributes such as `constraints.point_of_origin` and `radius`. Point of origin can be written as latitude/longitude or it can be written in a certain place. These two attribute must always be used together.

```
"constraints.point_of_origin": "|en: Red Crescent Sunrise
    Camp"
"constraints.radius_km": 10
```

- It should be noted in which regions the guardian relationship is legally valid as follows.

```
"constraints.jurisdictions": "|en: EU, India, Bangladesh"
```

- The condition of when the guardian relationship will be active must also be determined in the credential. This explanation should be written carefully and in a way that does not allow misunderstanding.

```
"constraints.trigger": "|en: Death of parent"
"constraints.circumstances": "|en: While a parent or
   adult sibling is unavailable, and no new guardian has
   been adjudicated."
```

- It can also be determined by the start and end dates when the relationship will be active. Date information should be written using the ISO8601 standard.

```
"constraints.startTime": "2019-07-01T18:00"
"constraints.endTime": "2019-08-01"
```

Finally, a audit trail is not mandatory but is strongly recommended. A record must be kept of every action the Guardian takes on behalf of the dependent. This record can be a document in JSON format as in the example below.

```
   "@type": "SIVPGTF audit/1.0",
   "event_time": "2019-07-25T18:03:26",
   "event_place": "@40.4043328,-111.7761829,15z",
   "challenger": "amy.smith@redcross.org",
   "witness": "fred.jones@redcross.org",
   "guardian": "Farooq Abdul Sami",
   "rationale": "natural parent",
   "dependent": "Isabel Sami, DOB 2009-05-21",
   "event": "enroll in class, receive books",
   "justifying_permissions": "school, necessaries"
   "evidence": // base64-encoded photo of Farooq and Isabel
```

In this way, we have shown the attributes required to create credentials for a guardian via use case. The full version of the credential schema definitions in this example can be seen in the Appendix A.

Chapter 4

Guardianship Use Case: Custody

In Chapter 3, we stated types, definitions and gave examples of indirect SSI control. Among these, Sovrin Guardianship Working Group has been working on several use cases. For instance, exemplary demo for the child refugee use case is shown over the Mya character [77]. Likewise, for an elderly person with dementia, the demo is described on the character Jamie [76]. In a similar way, we are now going to propose solutions for a guardianship use case, namely *custodianship*.

Custodianship is a relationship in which a person is legally cared for. Child custody is often the situation where the care and control of the child 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* [78] that is accepted by 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, to have an identity in her name and to be raised by a family or cultural group. It also states that even if the parents are separate, the child should be able to establish a relationship with both parents.
Various types of custody have emerged for different scenarios. These Custody types can be summarized as follows in [79].

1. **Sole Custody:** It is the most accepted and practised custody in the world. Only one parent has custody over the child. Although the other parent has the right to see the child on certain days, he or she has no right to speak legally in decisions that affect the child.

2. **Split Custody:** It is a type of custody in which the custody of children is shared among the parents in the case of more than one child. Generally, courts do not favour the separation of the siblings, but in some cases, this decision can be made. For example, when a single parent cannot care for all children.

3. **Divided Custody:** It is a type of custody in which a child's custody is shared among his family at certain times of the year. For example, during the school period, the child's custody is in the mother, but during the holiday period, the custody is in the father.

4. **Joint Custody:** It is the situation where the custody of the child is given to both parents. Both parents have the right to speak for situations that concern the child's life equally. At the same time, both parents have the right to access the child's legal records, such as medical records of the child. In some sources, it is referred to as "Joint Legal Custody" [79].

In this chapter, we are going to examine joint custody use case and search for possible solutions for this use case within self sovereign identity realm.

4.1 Joint Custody

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. Mother and father have equal voice over the child. Also, with international agreements, joint custody decision can be made as a result of divorces in most countries. After the divorce with joint custody, a child's parent has the right to speak equally in matters such as the care, protection, education and supervision of the child, just as before the divorce. Thus, neither the mother nor father has any advantage over each other.

Joint custody has many advantages for parents and children [79]. First of all, thanks to Joint custody, the concept of family continues even though marriages are ended as a result of divorces. Although the dynamics of the relationship between family members change, the continuity of the relationship is essential for the child's well-being. Various

researches such as [80] and [81] show that children get better outcomes as a result of joint custody and have more beneficial effects compared to sole custody. Another advantage is that it balances the power struggle between parents. Since both parents have equal rights over the child, harmful consequences arising from the power struggle between them will be prevented. It ensures that the family relationship that children need continues. It is also important for the continuity of the concept of father figure for a child. As a result, it is seen that joint custody has many advantages over sole custody in order to raise a mentally healthier and self-confident individual.

In order to make a joint custody decision as a result of the divorce, some conditions must be met.

- **The child's best interests:** This is the most important condition for the decision regarding joint custody. Even if the parties agree on every issue, if the child's best interests will be harmed, a joint custody decision is not made.
- **Volunteering:** Both mother and father need to volunteer on joint custody.
- **Approved by the judge:** It should be approved by the person who will make this decision.

4.1.1 Formalization of Joint Custody

A user story will be used to illustrate what problems the joint custody scenario causes in guardianship.

Charlie is a child who is the son of Alice and Bob. Alice and Bob decide to end their marriage. Upon the end of their marriage, both are ready to take on the guardianship of Charlie. But after divorcement procedures, the court decides to give *joint custody* for Charlie. This decision means that the management of Charlie's digital assets will have to be jointly controlled by both Alice and Bob until Charlie reached the age of majority. In a regular guardianship, it has one guardian and one dependent role. However, in this scenario, a dependent must have two guardians in the same relationship. In a proof verifying process to be performed on Charlie, the verifier will also need to verify the identity of Alice and Bob as guardians.

Within the above formalization, Charlie is dependent as a victim. Both Alice and Bob will be assumed to be malicious. That is, either of them can try to manipulate the protocol to gain any advantages. Joint custody will be in effect for a limited time until Charlie is over 18 years old. Naturally, court and SSI steward are assumed to be honest; thus, they will behave as trusted parties.

The characters and their stories in this formalization are summarized in Table 4.1

TABLE 4.1: User Story Summary of Our 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 Proof of Concept Use Case: Joint Custody

For the framework to be used in this use case, a competent authority is needed to 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. Later, local courts with the role of the issuer can give credentials using this framework.

In this framework, schema definition and credential definition must be defined before a credential can be issued. Schema definitions is a template that contains the information of attributes. Credential Definition has issuer's public key data for the 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.

4.2.1 Main Construction

As stated in Section 3.4, the 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 [82] on the Github page.

In order to create Custody framework, we should clearly state the name and versioning information.

```
"Trust Framework Name": "TR Custody Trust Framework"
"Version": 1.0
"Author": "It is maintained by the Ministry of Justice"
```

Then, the location where this framework is valid should be specified. In our use case, it specified as follows:

```
This framework is suitable for use by the courts within the
   boundaries of Turkey.
```

It should be clearly stated what metadata information should be in the credential and what its values may be. In custody framework, there is only one metadata, but additional metadata suggestions are also welcome.

```
"case-result": "sole-custody", "split-custody", "divided-
   custody", "joint-custody"
```

Then, we define the rationale of guardianship. Establishing this relationship can be one of 3 rationales, as shown below.

- `kinship`: If the person taking custody of the child is a relative, this should be chosen.
- `court-order`: This rationale is required for the legal enforcement of custody use case.
- `enforced`: If the person who takes custody of the child is not a relative, this should be chosen.

The required attributes for the identifying credential subject should be defined. Our framework defines the following ways to identify a Holder (guardian) in a credential:

- `first-name`: First name should be the name that the guardian acknowledges and answers to
- `last-name`: Last name may be empty if it is unknown
- `role`: It can be following values: `kinship`, `ad-hoc`, `legal_appointment`.
- `kinship-status`: If the role is a `kinship`, then this attribute should be filled. Otherwise, it can be left blank. Indicates the degree of kinship. Can be following values: `mother`, `father`, `grandmother`, `grandfather`, `uncle`, `other`.
- `rationalURI`: can be following values as described in rationales for Guardianship.

Our framework defines the following ways to identify a Proxied (dependent) in a credential:

- `first-name`: First name should be the name that the dependent acknowledges and answers to
- `last-name`: Last name may be empty if it is unknown
- `birth-date`: Birth date may be approximate
- `photo`: Photo is required and must be a color photo of at least 800x800 pixel resolution.

- `iris`: Not mandatory but strongly recommended.
- `fingerprint`: Not mandatory but strongly recommended.

Guardian's permissions retrieve data as a string list. Permissions have a special format called *Simple Grant Language (SGL)* which was created by Evernym Company [83]. SGL provides to create Role-Based Access Control mechanisms in JSON format. Guardians could have the following permissions:

- `routine-medical-care`: Includes all health related use case of dependent
- `school`: It is valid in school-related situations such as enrollment or choosing a course.
- `necessaries`: Basic vital needs
- `light-travel`: For day trips. Boarding stay is not covered.
- `extended-travel`: It covers long-term trips with boarding.
- `contracts`: Permission to issue legal documents on behalf of the Dependent.
- `delegate`: Being able to delegate an authority to another person.
- `successor`: It is the permission to choose a guardian that can replace after her.
- `authorize`: Authority to change permissions.

There must be certain constraints for Holder. These can be listed as follows.

- `boundaries`: Can be country, city or region. Boundaries are specified as a comma-separated list of strings.
- `pointOfOrigin`: Additional way to specify boundaries. It is a string that may use latitude/longitude notation
- `radiusKM`: Radius is an integer measured in kilometers. Must use with `pointOfOrigin`.
- `jurisdictions`: It indicates in which regions this relationship is valid legally.
- `trigger`:

- `circumstances`:
- `startTime`: Indicates on which date this relationship started. It is expressed as ISO8601 timestamps in UTC timezone.
- `endTime`: Indicates on which date this relationship ended. It is expressed as ISO8601 timestamps in UTC timezone.

Finally, it is strongly recommended that an audit trail be produced any time a guardian performs any action. Audit should be have following attributes: `event_time`, `event_place`, `guardian`, `dependent`, `event`, `justifying_permissions` and `evidence`.

The joint custody schema created using this framework can be seen in Appendix B. The framework's machine-readable JSON formatted version is in Appendix C. Now It will be demonstrated by implementing the credential creation and verification process using this framework.

4.2.2 Implementation

Using the framework created in the previous section, a court in the role of Issuer can now create credentials. But firstly, issuer creates a credential definition by reference to the JSON file, which is the machine-readable format of the framework. For Credential Definition, Issuer adds its digital signature and its own public key to the schema it will use, and publish it in the blockchain on the network. Thus, credentials that will be given using this definition will be verified by everyone.

When Alice and Bob come to court for divorce, the court will make a decision and create a custody credential for court result. 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. Complete credential can be seen in Appendix D.
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, Verifiable Organization Network (VON) will be used [84]. VON Network provides

```
[yusufdundar@Yusuf-MacBook-Pro ~> docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                                        NAMES
cc6ee43b39a6        von-network-base    "bash -c './scripts/…"   58 seconds ago      Up 57 seconds       0.0.0.0:9701-9702->9701-9702/tcp             von_node1_1
25e6b8e848d5        von-network-base    "bash -c 'sleep 10; …"   58 seconds ago      Up 57 seconds       0.0.0.0:9000->8000/tcp                       von_webserver_1
51549de21839        von-network-base    "bash -c './scripts/…"   58 seconds ago      Up 57 seconds       0.0.0.0:9705-9706->9705-9706/tcp             von_node3_1
06364ff628f4        von-network-base    "bash -c './scripts/…"   58 seconds ago      Up 57 seconds       0.0.0.0:9707-9708->9707-9708/tcp             von_node4_1
028404c064ae        von-network-base    "bash -c './scripts/…"   58 seconds ago      Up 57 seconds       0.0.0.0:9703-9704->9703-9704/tcp             von_node2_1
```

FIGURE 4.1: List of containers running on Docker

ledger browser on Docker environment. Ledger browser allows to examine transactions and see the status of nodes locally. Firstly we will make sure the Docker is running; then to start the VON Network we will run following commands on the terminal.

```
./manage build
./manage start --logs
```

We will make sure the network is up and running by controlling the Docker. Figure 4.1 shows that 4 node and web server are running properly. Thus, we can examine the status of the active nodes on the ledger via the browser. When the http://localhost:9000 address is entered from the browser, an information screen appears as shown in Figure 4.2. On the left side of this screen, you can see the states of nodes actively working on the network. Details of the transactions made on the network can be examined from the *Ledger State* section. As explained in Section 2.4.1, in the Ledger Section, it is seen that there are three different ledgers as expected. These can be summarized as follows.

1. **Domain Ledger**: where DID and Schema records are kept.
2. **Pool Ledger**: where node information is kept
3. **Config Ledger**: where network configuration is kept.

In the beginning, there are only five transactions. First transaction for the trustee DID, other four for Stewards that operate the four nodes.

Now 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 is verified via *present proof* protocol.

In this study, Swagger was used to observing the endpoints of the Agents easily. Swagger is an open-source tool that uses the OpenAPI specification [85]. Thanks to this tool, transactions performed on Agent endpoints can be viewed visually on the browser. First,

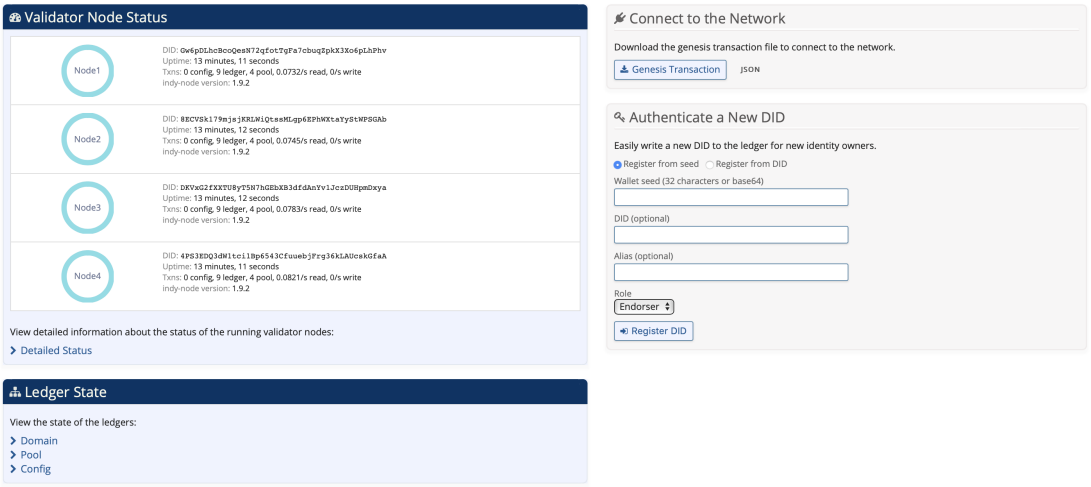


FIGURE 4.2: Nodes status screen over local VON Network

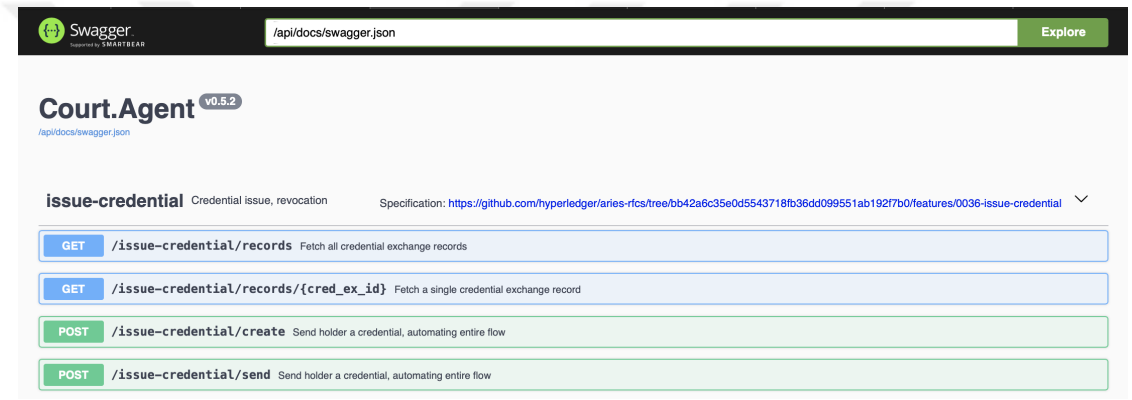


FIGURE 4.3: Court Agent Swagger User Interface on Browser

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 [86]. Court Agent's endpoint can be seen visually in Figure 4.3. Likewise, we make Alice's agent up and running. In this python script, we already registered Agents' DIDs into the public Indy ledger. Court Agent published the schema and credential definition, as seen in Figure 4.4. Also, it can 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 published the credential for Alice using the custody schema. This credential can be seen in Appendix E.

```
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 4.4: Court Agent's Schema for Custody

Chapter 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 [87] and matPlotLib [88] library. Thanks to this library, we were able to get the average duration of publishing guardian credentials graphically. In order to make an accurate measurement, we applied the *Straight Line Fitting* analysis method mentioned in [89].

To compare the Custody Framework we have created, we chose the Faber College Demo from the classic SSI samples that come 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 own Custody use case.
As a result, we obtained the graphic shown in Figure 5.1. Orange dots and green pluses show the measured results in ms. The slope of the straight lines shown in the graph give us the average time. Accordingly, the average time for the Faber Demo 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 much more attribute. The difference is negligible. The result also same for the verification process as seen in Figure 5.2. Thus, the result we obtained from these

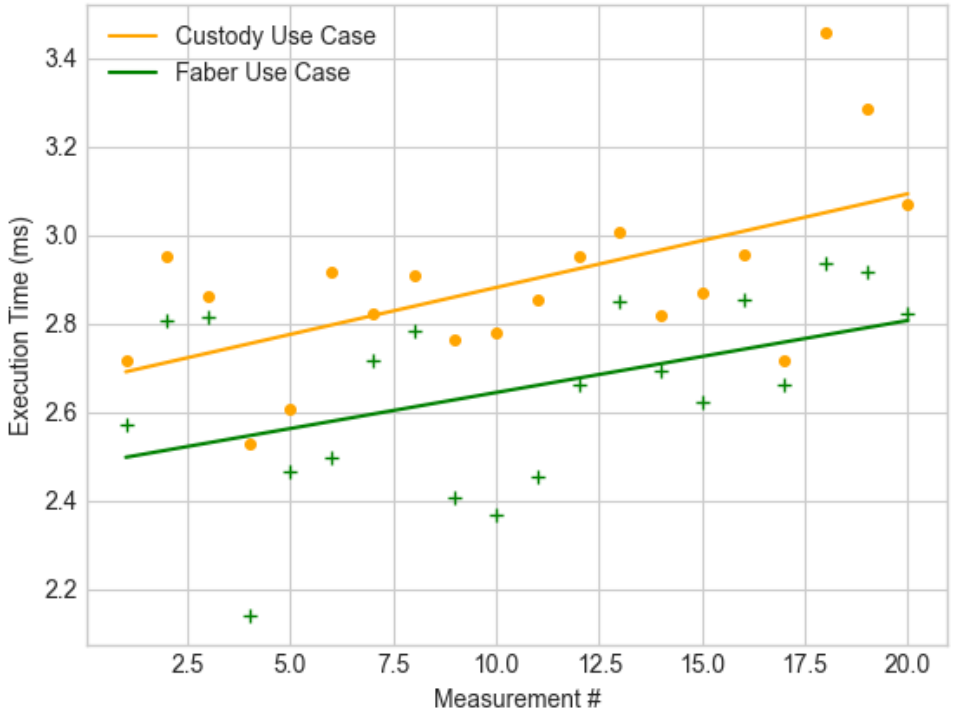


FIGURE 5.1: Comparison between Faber Use Case and Custody Use Case

measurements shows that the custody framework does not bring too much extra load to existing systems.

The point to be careful here is that we took only the part of making the credential schema into credential offer in order not to be affected by the delays caused by network. We did not include the function of publishing this credential to Ledger on this measure. However, the measurement results, including the publication of credential to ledger, are also shown in Table 5.1.

TABLE 5.1: 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,

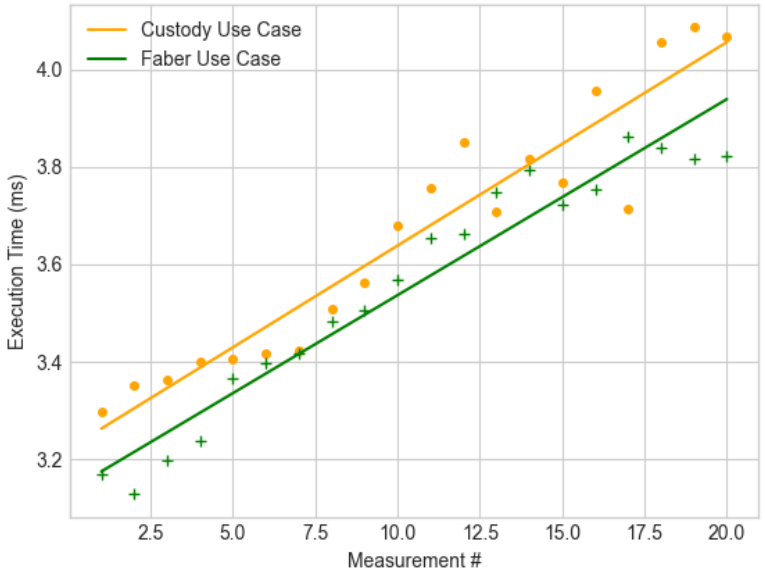


FIGURE 5.2: Verify the proof time comparison between Faber Use Case and Custody Use Case

it should be able to meet every real-life scenario. Therefore, the guardianship is of great importance. Because there are many people who cannot manage their own digital data in real life. In order 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 can work with the existing infrastructure. In this way, we have shown that the mutual guardian relationship scenario, which will surely be encountered in real life, is not an obstacle in the adoption of 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 as described in Section 2.4.5. Therefore, while looking for a solution to this problem, we should consider the possible solutions separately on these different layers. Although we have shown that the process of realizing the joint custody scenario by changing the credential attributes as described Chapter 4, different solutions can be considered for future studies.

Chapter 6

Conclusion

This work started with the definition of *Identity*. Then, the concepts of Physical identity and Digital identity were explained and compared. The history and types of digital identities are briefly explained. In the current situation, chronic problems experienced in digital identities have been identified and continued to work with Self Sovereign Identity, which can be a solution to these problems. The technologies used by SSI are explained in detail in Chapter 2. The principles of SSI are listed with references. Existing SSI studies are mentioned and compared with other systems. After explaining the technologies used by SSI in its infrastructure, Indirect Identity Control systems, the main subject of the study, started to be explained. Due to the wide usage area of SSI, Indirect SSI Control subtitle was chosen as the subject to be studied. Indirect SSI Control is a new subject that various working groups work on a proof of concept of the system through use cases. Indirect identity control types are explained item by item and their importance in SSI systems is stated.

Then, Guardianship topic was explained in detail in Chapter 3. We went over the question of how digital identities can be managed with the mutual guardianship. Since the relation of Guardianship was a new concept in SSI there was not much work on the subject of mutual guardianship. Groups working on the Guardianship and their work are mentioned. The Guardianship life cycle is described and its risks and challenges itemized. Afterwards, a sample model for guardianship in existing systems is examined

and explained how to implant through a use case.

Finally, the topic of mutual guardianship and the problems it has created in existing systems have been mentioned and a model has been designed to solve these problems in Chapter 4. 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 present proof, thus ensuring that the operation is working properly within the existing SSI structure. After making sure that the Custody Framework created for Mutual guardianship is working properly, 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 a joint custody of their children as a result of divorce. In this case, what kind of differences should be done in the infrastructure of digital identities has been researched, a model has been created and a new way about mutual guardianship has been opened for further studies.

As a future work, it is necessary to examine how this system will work in other scenarios that we may encounter in real life. As studied by Crites *et al.* in [90], a solution can be developed by changing credentials cryptographic infrastructure. Delegatable credentials can be created by changing the signing scheme used. We will leave these works for future projects as it extends scope of this thesis. Also other studies such as [54] and [91] can be taken as a reference for future studies.

It should be noted that while applying different solutions, the performance of the system should not be affected. It should also be looked at how compatible the new solutions are with the existing system. One of the biggest obstacles to mass adoption is that

it is incompatible with existing systems. At the same time, a solution to be made by disrupting or complicating the usage habits of the user will not be applicable.

Appendix A

Credential Schema Example

A sample schema is defined for credentials to be created for guardianship in [92] with JSON format as shown in below.

```
[
  "@context", //JSON array where first = uri w3.org/2018/
   credentials/v1
  "id", // optional
  "type", //"[\",VerifiableCredential\", \"Guardianship/1.0\"
   ]",
  "credentialSchema", // points to cred def
  "issuanceDate", // standard in VCs RFC3339 Date-Time string
  "expirationDate",
  "credentialStatus", { "id" /*points to rev reg*/, "type": "
   sovrin rev reg 1.0" }
  "issuer", // <did>
  "trustFramework", // uri to a website
  "auditURI", // constrained by trust framework, but a URI
  "appealURI", // same
  "credentialSubject.holder.type", // uri to schema for a
   holder
  "credentialSubject.holder.role", // "parent" or "
   grandparent"
```

```
15   "credentialSubject.holder.rationaleURI", // uri from trust
      framework that defines which rationale justifies the
      guardianship
16   "credentialSubject.holder.firstName",
17   "credentialSubject.holder.constraints.boundaries",
18   "credentialSubject.holder.constraints.pointOfOrigin",
19   "credentialSubject.holder.constraints.radiusKM",
20   "credentialSubject.holder.constraints.jurisdictions",
21   "credentialSubject.holder.constraints.trigger",
22   "credentialSubject.holder.constraints.circumstances",
23   "credentialSubject.holder.constraints.startTime",
24   "credentialSubject.holder.constraints.endTime",
25   "credentialSubject.proxied.type", // uri to schema for a
      dependent
26   "credentialSubject.proxied.permissions", // string of JSON
      array of grants
27   "credentialSubject.proxied.firstName",
28   "credentialSubject.proxied.lastName",
29   "credentialSubject.proxied.birthDate",
30   "credentialSubject.proxied.gender",
31   "credentialSubject.proxied.nativeLanguage",
32   "credentialSubject.proxied.identifyingMarks",
33   "credentialSubject.proxied.photo",
34   "credentialSubject.proxied.iris",
35   "credentialSubject.proxied.fingerprint"
36 ]
```

Appendix B

Custody Schema

```
[
  "@context",
  "id",
  "type",
  "credentialSchema",
  "issuanceDate",
  "expirationDate",
  "credentialStatus", { "id", "type" }
  "issuer",
  "trustFrameworkURI",
  "auditURI",
  "appealURI",
  "caseResult",
  "credentialSubject.holder.type",
  "credentialSubject.holder.role",
  "credentialSubject.holder.rationaleURI",
  "credentialSubject.holder.firstName",
  "credentialSubject.holder.lastName",
  "credentialSubject.holder.kinshipStatus",
  "credentialSubject.holder.constraints.boundaries",
  "credentialSubject.holder.constraints.pointOfOrigin",
  "credentialSubject.holder.constraints.radiusKM",
```

```
  "credentialSubject.holder.constraints.jurisdictions",
  "credentialSubject.holder.constraints.trigger",
  "credentialSubject.holder.constraints.circumstances",
  "credentialSubject.holder.constraints.startTime",
  "credentialSubject.holder.constraints.endTime",
  "credentialSubject.proxied.type",
  "credentialSubject.proxied.firstName",
  "credentialSubject.proxied.lastName",
  "credentialSubject.proxied.birthDate",
  "credentialSubject.proxied.photo",
  "credentialSubject.proxied.iris",
  "credentialSubject.proxied.fingerprint"
  "credentialSubject.proxied.permissions",
]
```

Appendix C

Custody Trust Framework

```
{
  "@context": "https://github.com/yusufdundar/2020-MScThesis/blob/master/context.jsonld",
  "name": "TR Custody Trust Framework",
  "version": "1.0",
  "author": "Ministry of Justice"
  "description": "Custody Framework for Courts"
  "docs_uri": "https://github.com/yusufdundar/2020-MScThesis/blob/master/custody-framework.md",
  "data_uri": "https://github.com/yusufdundar/2020-MScThesis/custody-trust-fw.json",
  "topics": ["Custody"],
  "jurisdictions": ["tur"],
  "geos": ["Turkey"]
  "case_result": ["sole-custody", "split-custody", "divided-custody", "joint-custody"],
  "identifying": [
        {"Holder": [
            "first-name",
            "last-name",
            "role": ["kinship", "ad-hoc", "legal_appointment"
  ],
```

```
            "kinship-status": ["mother", "father", "
grandmother", "grandfather", "uncle", "other"],
            "rationalURI": ["kinship","court-order","enforced
"]
        ]},
        {"Proxied": [
            "first-name",
            "last-name",
            "birth-date",
            "photo",
            "iris",
            "fingerprint"
        ]},
    ],
    "permissions": [
      "routine-medical-care",
      "school",
      "necessaries",
      "light-travel",
      "extended-travel",
      "contracts",
      "delegate",
      "successor",
      "authorize"
    ],
    "constraints": [
      "boundaries",
      "pointOfOrigin",
      "radiusKM",
      "jurisdictions",
      "trigger",
      "circumstances",
      "startTime",
```

```
      "endTime"
    ],
    "define": [
        {"name": "MOJ": "id": "did:example:abc123"},
        {"name": "TCG": "id": "did:anotherexample:def456"},
    ],
   "audit": {"uri": "https://www.adalet.gov.tr/audit"}
}
```

Appendix D

Custody Credential

```
1
2  {
3    "@context": [
4      "https://www.w3.org/2018/credentials/v1",
5      "https://github.com/yusufdundar/2020-MScThesis/blob/
       master/context.jsonld"
6    ],
7    "id": "http://example.com/credentials/3732",
8    "type": ["VerifiableCredential", "Proxy.G/TR Custody Trust
       Framework/1.0/JointCustody"],
9    "credentialSchema": {
10     "id": "did:example:cdf:35LB7w9ueWbagPL94T9bMLtyXDj9pX5o",
11     "type": "did:example:schema:22
       KpkXgecryx9k7N6XN1QoN3gXwBkSU8SfyyYQG"
12   },
13   "issuer": "https://moj.gov/issuers/14",
14   "issuanceDate": "2020-01-01T19:23:24Z",
15
16   "trustFrameworkURI": "https://github.com/yusufdundar/2020-
       MScThesis/blob/master/custody-framework.md",
17   "auditURI": "https://example.org/audit",
18   "appealURI": "https://example.org/appeal",
```

```
19
20  "credentialSubject": {
21    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
22    "holder": {
23      "type": "Holder",
24      "firstName": "Alice",
25      "lastName": "Smith"
26      "role": "kinship",
27      "kinshipStatus": "mother"
28      "rationaleURI": "court-order",
29      "constraints": {
30        "type": "Constraints",
31        "boundaries": "Within Turkey country limits",
32        "pointOfOrigin": "Ankara",
33        "radiusKM": 1000,
34        "jurisdictions": "tur",
35        "trigger": "|en: Death of parent",
36        "circumstances": "|en: While a parent or adult
   sibling is unavailable , and no new guardian hasbeen
   adjudicated .",
37        "startTime": "25 -07 -01 T18 :00",
38        "endTime": "2025 -08 -01"
39      }
40    },
41    "proxied": {
42      "type": "Proxied",
43      "firstName": "Charlie"
44      "lastName": "Smith"
45      "birthDate": "2014-08-01"
46      "photo": "https://raw.githubusercontent.com/yusufdundar
   /2020-MScThesis/master/charlie.base64"
47      "iris":
48      "fingerprint":
```

```
      "permissions": [
        {
          "grant": ["routine-medical-care"],
          "when": {"any": [
            {"kinshipStatus": "mother"},
            {"kinshipStatus": "father"}
            ]
          }
        },
        {
          "grant": ["school"],
          "when": {"all": [
            {"n":2, "role": "kinship"}
            ]
          }
        }

      ],
    }
  },
  "credentialStatus": {
    "id": "https://example.edu/status/24",
    "type": "CredentialStatusList2020"
  }
  "proof": {
    "type": "CLSignature2019",
    "issuerData": "5NQ4TgzNfSQxoLzf2d5AV3JNiCdMaTgm...
   BXiX5UggB381QU7ZCgqWivUmy4D",
    "attributes": "pPYmqDvwwWBDPNykXVrBtKdsJDeZUGFA...
   tTERiLqsZ5oxCoCSodPQaggkDJy",
    "signature": "8eGWSiTiWtEA8WnBwX4T259STpxpRKuk...
   kpFnikqqSP3GMW7mVxC4chxFhVs",
```

```
78      "signatureCorrectnessProof": "
        SNQbW3u1QV5q89qhxA1xyVqFa6jCrKwv...
        dsRypyuGGK3RhhBUvH1tPEL8orH"
79    }
80  }
```

Appendix E

Credential Issued by Court Agent


```
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": "42038558799018855740133663747148856923173233228163099647287684958824673059021361552303484052638014179026855791681990525308118392198991420103350136329176491963827010888330809791533879902727885550092775118508720725659832118594971908737068675882949407498440378256631170570022654602887469405336683894334394156460479998322115445665542382440249650850737742152070981705653252701478153378616235628357564748210985025952998854254798958448035479643129740569466106167987498579060046372328855617351328775631648814819217127229478574984251851106705425923773010601116289488054201685195116403310560799932929345025990051430349448118202743007086797934881940019",
      "vr_prime": null
    },
    "nonce": "246372910291030255120242",
    "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 E.1: Credential Issued by Court Agent

Bibliography

[1] Alexander Mühle, Andreas Grüner, Tatiana Gayvoronskaya, and Christoph Meinel. A survey on essential components of a self-sovereign identity. *Computer Science Review*, 30:80 – 86, 2018. ISSN 1574-0137. doi: https://doi.org/10.1016/j.cosrev.2018.10.002. URL `http://www.sciencedirect.com/science/article/pii/S1574013718301217`.

[2] David Chadwick Manu Sporny, Dave Longley. Verifiable credentials data model 1.0, November 2019. URL `https://www.w3.org/TR/vc-data-model/`. [Online].

[3] Drummond Reed. Decentralized identifiers (dids): The fundamental building block of self-sovereign identity (ssi), May 2018. URL `https://www.slideshare.net/SSIMeetup/decentralized-identifiers-dids-the-fundamental-building-block-of-selfsovereign-identity-ssi`. [Online].

[4] Imran Bashir. *Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained*. Packt Publishing Ltd, 2018.

[5] Drummond Reed, Jason Law, and Daniel Hardman. The technical foundations of sovrin. Technical report, Sovrin, 2016. Retrieved from: https://www.evernym.com/wp-content/uploads/2017/07/The-Technical-Foundations-of-Sovrin.pdf, 2016.

[6] Matthew Davie, Dan Gisolfi, Daniel Hardman, John Jordan, Darrell O'Donnell, and Drummond Reed. The trust over ip stack. *IEEE Communications Standards Magazine*, 3(4):46–51, 2019.

[7] Nathan George. Hyperledger Aries: Open Source Interoperable Identity Solution, June 2019. URL `https://ssimeetup.org/hyperledger-aries-open-source-interoperable-identity-solutions-nathan-george-webinar-30/`. [Online].

[8] Sovrin Guardianship Task Force. On guardianship in self-sovereign identity, December 2019. URL `https://sovrin.org/wp-content/uploads/Guardianship-Whitepaper2.pdf`. [Online; posted 25-December-2019].

[9] Daniel Hardman. Aries rfc 0103: Indirect identity control, June 2019. URL `https://github.com/hyperledger/aries-rfcs/tree/master/concepts/0103-indirect-identity-control`. [Online].

[10] SSI in IoT Team Sovrin Trust Framework Working Group ID4All Team. Persona & user stories, April 2018. URL `https://docs.google.com/presentation/d/1qUYQa7U1jczEFun3a7sB3lKHIprlwd7brfOU9hEJ34U/edit#slide=id.p7`. [Online].

[11] Andrew Tobin and Drummond Reed. The inevitable rise of self-sovereign identity. *The Sovrin Foundation*, 29, 2016.

[12] Alex Andrade-Walz. What are decentralized identifiers (dids)?, September 2019. URL `https://www.evernym.com/blog/what-are-decentralized-identifiers-dids`. [Online].

[13] Sovrin Foundation. Sovrin glossary v3, December 2019. URL `https://docs.google.com/document/d/1gfIz5TT0cNp2kxGMLFXr19x1uoZsruUe_0glHst2fZ8/edit`. [Online].

[14] Derrick Rountree. *Federated identity primer*. Newnes, 2012.

[15] Phillip J Windley. *Digital Identity: Unmasking identity management architecture (IMA)*. " O'Reilly Media, Inc.", 2005.

[16] Christopher Allen. The path to self-sovereign identity, April 2016. URL `http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html`. [Online; posted 25-April-2016].

[17] Audun Jøsang and Simon Pope. User centric identity management. In *AusCERT Asia Pacific information technology security conference*, page 77. Citeseer, 2005.

[18] Eve Maler, Conor P Cahill, AOL John Hughes, Michael Beach, Boeing Rebekah Metz, Rick Randall, Thomas Wisniewski, Entrust Irving Reid, Paula Austel, Maryann Hondo, et al. Security and privacy considerations for the oasis security assertion markup language (saml) v2. 0. *Language (SAML)*, 2:0, 2005.

[19] Liberty Alliance Project. Liberty id-ff architecture overview version: 1.2-errata-v1.0, 2005. URL http://www.projectliberty.org/liberty/content/download/318/2366/file/draft-liberty-idff-arch-overview-1.2-errata-v1.0.pdf. [Online].

[20] IIW. Main page — iiw,, 2020. URL https://iiw.idcommons.net/index.php?title=Main_Page&oldid=22716. [Online; accessed 31-May-2020].

[21] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, and C. Mortimore. Openid connect core 1.0. Technical report, OpenID Foundation, 11 2014. URL https://openid.net/specs/openid-connect-core-1_0.html#Introduction.

[22] Dick Hardt et al. The oauth 2.0 authorization framework. Technical report, RFC 6749, October, 2012.

[23] Sovrin Foundation. What is self-sovereign identity?, December 2018. URL https://sovrin.org/faq/what-is-self-sovereign-identity/. [Online].

[24] Muneeb Ali, Jude Nelson, Ryan Shea, and Michael J Freedman. Blockstack: A global naming and storage system secured by blockchains. In *2016 Annual Technical Conference 16*, pages 181–194, 2016.

[25] Christian Lundkvist, Rouven Heck, Joel Torstensson, Zac Mitton, and Michael Sena. Uport: A platform for self-sovereign identity. *URL: https://whitepaper.uport.me/uPort_whitepaper_DRAFT20170221.pdf*, 2017.

[26] Phil Windley. Fixing the five problems of internet identity, October 2017. URL https://www.windley.com/archives/2017/10/fixing_the_five_problems_of_internet_identity.shtml. [Online].

[27] Security.org. Public awareness of major data breaches, 2018. URL https://www.security.org/resources/data-breach-awareness/. [Online].

[28] Todd Spangler. Yahoo says 3 billion user accounts were hacked, upping previous estimate, October 2017. URL https://finance.yahoo.com/news/yahoo-says-3-billion-user-205330487.html. [Online].

[29] Samuel Gibbs. Dropbox hack leads to leaking of 68m user passwords on the internet, August 2016. URL https://www.theguardian.com/technology/2016/aug/31/dropbox-hack-passwords-68m-data-breach. [Online].

[30] Robert Hackett. Linkedin lost 167 million account credentials in data breach, May 2016. URL https://fortune.com/2016/05/18/linkedin-data-breach-email-password/. [Online].

[31] Heather Adkins. An update on attempted man-in-the-middle attacks, August 2011. URL https://security.googleblog.com/2011/08/update-on-attempted-man-in-middle.html. [Online].

[32] Drummond Reed, Manu Sporny, Dave Longley, Christopher Allen, Ryan Grant, and Markus Sabadello. Decentralized identifiers (dids) v1.0, April 2020. URL https://www.w3.org/TR/did-core/. [Online].

[33] Moxie Marlinspike. What is "sovereign source authority"?, February 2012. URL https://www.moxytongue.com/2012/02/what-is-sovereign-source-authority.html. [Online].

[34] Satoshi Nakamoto and A Bitcoin. A peer-to-peer electronic cash system. *Bitcoin.–URL: https://bitcoin.org/bitcoin.pdf*, 2008.

[35] Natarajan, Harish Krause, Solvej Gradstein, and Helen. *Distributed Ledger Technology and Blockchain*. World Bank, 2017. doi: 10.1596/29053. URL https://elibrary.worldbank.org/doi/abs/10.1596/29053.

[36] Hyperledger. Hyperledger indy documentation v1.0, - 2018. URL https://hyperledger-indy.readthedocs.io/en/latest/index.html. [Online].

[37] The Linux Foundation. Hyperledger Ursa, - 2018. URL https://www.hyperledger.org/use/ursa. [Online].

[38] Nathan George. Announcing Hyperledger Aries, infrastructure supporting interoperable identity solutions, May 2019. URL https://www.hyperledger.org/blog/2019/05/14/announcing-hyperledger-aries-infrastructure-supporting-interoperable-identity-solutions. [Online].

[39] Sovrin Foundation. Sovrin governance framework, - -. URL https://sovrin.org/library/sovrin-governance-framework/. [Online].

[40] British Columbia. Orgbook bc, 2019. URL https://orgbook.gov.bc.ca/en/home. [Online].

[41] Paul Dunphy, Luke Garratt, and Fabien Petitcolas. Decentralizing digital identity: Open challenges for distributed ledgers. In *2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 75–78. IEEE, 2018.

[42] Uwe Der, Stefan Jähnichen, and Jan Sürmeli. Self-sovereign identity − opportunities and challenges for the digital revolution. *arXiv preprint arXiv:1712.01767*, 2017.

[43] Andreas Abraham. Self-sovereign identity, 2017.

[44] Marvin van Wingerde. *Blockchain-enabled self-sovereign identity.* PhD thesis, Master's thesis, Tilburg University, School of Economics and Management, 2017.

[45] Dirk van Bokkem, Rico Hageman, Gijs Koning, Luat Nguyen, and Naqib Zarin. Self-sovereign identity solutions: The necessity of blockchain technology. *arXiv preprint arXiv:1904.12816*, 2019.

[46] civic.com. Civic - our missions, 2017. URL `https://www.civic.com/company/`. [Online].

[47] id2020.org. The alliance manifesto. URL `https://id2020.org/manifesto`. [Online].

[48] Timothy Ruff. Introducing self-sovereign student id, August 2020. URL `https://medium.com/@rufftimo/introducing-self-sovereign-student-id-part-1-of-2-d3104ab0f8f2`.

[49] Christopher Allen and Shannon Appelcline. Smartcustody: Use of advanced cryptographic tools to improve the care, maintenance, control, and protection of digital assets, March 2019. URL `https://www.smartcustody.com/`.

[50] Phillip J Windley. How sovrin works. *Windley.com*, 2016.

[51] Jan Camenisch and Els Van Herreweghen. Design and implementation of the idemix anonymous credential system. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 21–30, 2002.

[52] Galia Kondova and Jörn Erbguth. Self-sovereign identity on public blockchains and the gdpr. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pages 342–345, 2020.

[53] Daan Weller and Raoul Dijksman. Blockchain's relationship with sovrin for digital self-sovereign identities. 2019.

[54] Jan Camenisch, Manu Drijvers, and Maria Dubovitskaya. Practical uc-secure delegatable credentials with attributes and their application to blockchain. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 683–699, 2017.

[55] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman & Hall/CRC, 2nd edition, 2014. ISBN 1466570261, 9781466570269.

[56] Jonathan E Stern. The electronic signatures in global and national commerce act. *Berkeley Technology Law Journal*, pages 391–414, 2001.

[57] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21 (2):120–126, 1978.

[58] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *International Conference on Security in Communication Networks*, pages 268–289. Springer, 2002.

[59] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208, 1989.

[60] Paul Leach, Michael Mealling, and Rich Salz. A universally unique identifier (uuid) urn namespace. 2005.

[61] Tim Berners-Lee, Roy Fielding, Larry Masinter, et al. Uniform resource identifiers (uri): Generic syntax, 1998.

[62] Markus Sabadello Drummond Reed. Decentralized identifier (dids) fundamentals deep dive, May 2018. URL `https://ssimeetup.org/decentralized-identifiers-dids-fundamentals-identitybook-info-drummond-reed-markus-sabadello-webinar-46/`. [Online].

[63] David Chaum. Blind signatures for untraceable payments. In *Advances in cryptology*, pages 199–203. Springer, 1983.

[64] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *International conference on the theory and applications of cryptographic techniques*, pages 93–118. Springer, 2001.

[65] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *Annual International Cryptology Conference*, pages 56–72. Springer, 2004.

[66] Jan Camenisch and Thomas Groß. Efficient attributes for anonymous credentials. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 345–356, 2008.

[67] Dmitry Khovratovich Michael Lodder, Brent Zundel. Pairings-based anonymous credentials with circuit-based revocation and permission policies. 2019.

[68] Aiden Slavin. Distributed ledger identification systems in the humanitarian sector. 2019.

[69] Mariano Belinky, Emmet Rennick, and Andrew Veitch. The fintech 2.0 paper: Rebooting financial services. *Santander InnoVentures/Oliver Wyman/Anthemis*, 2015.

[70] Pierre-Louis Aublin, Sonia Ben Mokhtar, and Vivien Quéma. Rbft: Redundant byzantine fault tolerance. In *2013 IEEE 33rd International Conference on Distributed Computing Systems*, pages 297–306. IEEE, 2013.

[71] Hyperledger. Sample guardianship trust framework, 2019. URL `https://github.com/hyperledger/aries-rfcs/blob/master/concepts/0103-indirect-identity-control/guardianship-sample/trust-framework.md`. [Online].

[72] Hyperledger. Hyperledger aries cloud agent - python, - -. URL `https://github.com/hyperledger/aries-cloudagent-python`. [Online].

[73] Hodl Helper Adrian Doerk. An introduction to self-sovereign identity. (SSI), October 2019. URL `https://medium.com/@SSI_Ambassador/an-introduction-to-self-sovereign-identity-ssi-916eb42f0490`. [Online].

[74] Simon Kemp. Digital 2019: Global digital overview, January 2019. URL https://datareportal.com/reports/digital-2019-global-digital-overview. [Online].

[75] Sovrin Foundation. Sovrin governance framework v2, December 2019. URL https://sovrin.org/wp-content/uploads/Sovrin-Governance-Framework-V2-Master-Document-V2.pdf. [Online].

[76] Sovrin Guardianship Taskforce. Concept demo: Guardianship for dementia, 2019. URL https://drive.google.com/file/d/1bxE1DNxN8OgkQrCbkJzBRzUsOf7KcjH2/view. [Online].

[77] Sovrin Guardianship Taskforce. Concept demo: Guardianship, 2019. URL https://drive.google.com/file/d/10sfYKp6Ohi_rLsNqb1GBrhuE0IuoBX2k/view. [Online].

[78] UN General Assembly. Convention on the rights of the child. *United Nations, Treaty Series*, 1577(3), 1989.

[79] Jay Folberg. *Joint custody and shared parenting*. Guilford Press, 1991.

[80] Robert Bauserman. Child adjustment in joint-custody versus sole-custody arrangements: a meta-analytic review. *Journal of Family psychology*, 16(1):91, 2002.

[81] Marjorie Lindner Gunnoe and Sanford L Braver. The effects of joint legal custody on mothers, fathers, and children controlling for factors that predispose a sole maternal versus joint legal award. *Law and human behavior*, 25(1):25–43, 2001.

[82] Yusuf Dündar. 2020-MScThesis-Custody Trust Framework, June 2020. URL https://github.com/yusufdundar/2020-MScThesis.

[83] Daniel Hardman. Sgl (simple grant language), June 2019. URL https://evernym.github.io/sgl/docs/tutorial.html. [Online].

[84] Province of British Columbia. VON Network, April 2018. URL https://github.com/bcgov/von-network.

[85] https://swagger.io/. OpenAPI Specification, February 2020. URL https://swagger.io/specification/.

[86] Yusuf Dündar. 2020-MScThesis-Custody Trust Framework, June 2020. URL https://github.com/yusufdundar/2020-MScThesis/tree/master/src.

[87] https://numpy.org. The fundamental package for scientific computing with Python, - 2006. URL `https://numpy.org`.

[88] https://matplotlib.org. Matplotlib: Visualization with Pythonn, - 2003. URL `https://matplotlib.org`.

[89] Carlos Moreno and Sebastian Fischmeister. Accurate measurement of small execution times getting around measurement errors. *IEEE Embedded Systems Letters*, 9 (1):17–20, 2017.

[90] Elizabeth C Crites and Anna Lysyanskaya. Delegatable anonymous credentials from mercurial signatures. In *Cryptographers' Track at the RSA Conference*, pages 535–555. Springer, 2019.

[91] Johannes Blömer and Jan Bobolz. Delegatable attribute-based anonymous credentials from dynamically malleable signatures. In *International Conference on Applied Cryptography and Network Security*, pages 221–239. Springer, 2018.

[92] Hyperledger. Sample guardianship trust framework, 2019. URL `https://github.com/hyperledger/aries-rfcs/blob/master/concepts/0103-indirect-identity-control/guardianship-sample/schema.json`. [Online].

Yusuf Dündar

Tuzla, Istanbul, ZC 34400
http://yusufdundar.com

Education

Marmara University M.Sc. Cybersecurity Engineering
Sept, 2017 - Present
Yıldız Technical University B.Sc. Electronic and Communication Engineering
Sept, 2011 - June, 2016
Bursa Science High School
Sept, 2006 - June, 2010

Work Experience

Researcher Dec, 2015 - Present
National PKI Lab. TUBITAK BILGEM
Trainee Aug, 2015 - Sept, 2015
Gdansk University of Technology, Poland

Publications

- Yusuf Dündar, and İsa Sertkaya. "Self sovereign identity based mutual guardianship" Journal of Modern Technology and Engineering [2020] Vol.5 / No.3 189-211.
- Yusuf Dündar, and Revna Acar Vural "Embedded system based strategical positioning with RFID." 2016 National Conference on Electrical, Electronics and Biomedical Engineering (ELECO).

Research Interests

- Cryptography
- Public Key Infrastructure
- Distributed Systems