Thesis - Field tested applications of Self-Sovereign Identity

Tim Speelman - 4096533

April 18, 2020

Preface

Status: *Placeholder*

Thanks.

Contents

1 | Introduction

Status: *Placeholder*

TODO: *Flesh out these bullets.*

TODO: *Introduce term Identity, which in itself is vague*

1.

1.1 Identity is everywhere

Status: *Placeholder*

1. We use identity every day of our lives (in any domain, many purposes, both physical and digital)
2. We have many means to identify ourselves (wallets filled with cards, biometrics, passwords, SIM cards)
3. Others also means to identify us, to track our movements and profile us (tracking cookies, facial recognition, bonus cards)
4. This enables us to do many things: communicate, pay and get paid, ..

TODO: *Walk through an identity scenario, airport security. What really happens there? Identification, identity linking, credentials, trust anchor (trusted third party), attribute verification, revocation check, verifier knows how to verify?, subject and issuer (add pic of my passport)*

TODO: *Optional: introduce Eve here? Forgeability*

TODO: *Compare that scenario to others. Which elements are recurring? Which are different?*

TODO: *Then move to digital, what is different?*

1.2 Identity needs recognition

Status: *Placeholder*

TODO: *Legally Enabled Identity, developing countries, physical identity system as basis for digital*

1.

1.3 The Evolution of Digital Identity Systems

Status: *Placeholder*

1. When internet became a thing, it did not come with identity built in.
2. So, individual companies started making individual identity systems.

3. Yet these systems are expensive to create, run and protect which made it a hurdle for new services.
4. Simultaneously, as digital life grew, people had difficulty coping with all their different accounts.
5. So, identity federation came into existence: allowing people to have an account with one provider, but use it at many others.
6. This reduced the effort both for service providers as well as users. But what about the cost?
7. The federation owner now had all the cost, but many offered their services for free.
8. Their business model clearly shows the disadvantage of federation: it allowed them to make elaborate profiles of users, which are used for targeted advertising, political or social manipulation or other purposes.
9. This is how the greedy lured in the sloths.
10. TODO: *User centric, MS passport, web of trust*

1.4 The Need for a Shared Infrastructure

Status: *Placeholder*

1. The need for sharing identity cost became clear.
2. However,

1.5 Self-Sovereign Identity, a new paradigm

Status: *Placeholder*

TODO: *Write*

1.6 Research Questions

Main research question

> How can one practically verify an actor's claims using a scalable Self-Sovereign Identity infrastructure?

TODO: *Rephrase RQ to include representation/delegation element*

Subquestions:

1. What mininum-viable constraints are imposed by the Self-Sovereign Identity paradigm?
2. How can a verifier rely upon information provided by a subject, even when he does not know or trust the issuer of that information directly?
3. How do we connect (existing) information systems?

4. How can a user have smart phone based control over his identities, their data and signatures.
5. How can a user authorize or delegate responsibility to another user?

1.7 Document Structure

Status: *Placeholder*

TODO: *Write*

First Law	User Control and Consent
Second Law	Minimal Disclosure for a Constrained Use
Third Law	Justifiable Parties
Fourth Law	Directed Identity
Fifth Law	Pluralism of Operators and Technologies
Sixth Law	Human Integration
Seventh Law	Consistent Experience Across Contexts

Table 1.1: Laws of Identity (Kim Cameron)

2 | Problem Description

Status: *Done*

The goal of this thesis project is to realize practical verification of an actor's qualifications on a scalable Self-Sovereign Identity infrastructure. Consider the following abstract scenario:

> Alice and Bob engage in some transaction in or after which Alice's behaviour may be harmful to Bob. Bob's uncertainty of Alice's future behaviour puts him at risk. To minimize his risk, Bob must gain confidence that Alice *qualifies* for the intended transaction. Therefore, Bob relies on *claims* about Alice made by another actor, Chris.
>
> If Chris's claims may somehow be *false*, Bob is again at risk. So Bob must either *trust* Chris with this, or gain confidence that Chris himself qualifies for making such claims. For this, Bob relies on claims by yet another actor, making the problem recursive.

Figure 2.1 illustrates this scenario with Alice as Subject, Bob as Verifier of Claims made by Chris, the Issuer. In the second part of the scenario, Chris plays the role of Subject [1].

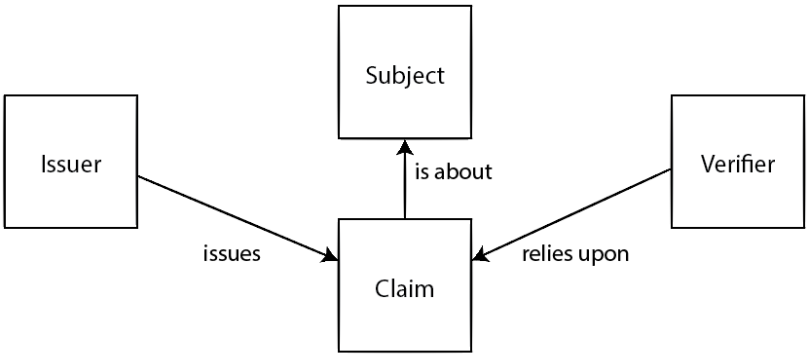


Figure 2.1: Semantic Model of Claim Based Identity

As argued in the introduction, a (self-sovereign) identity infrastructure should support a multitude of use cases. Each *use case* is an instance of the abstract scenario described above. In particular, such an instance describes:

- Which *transaction* Alice and Bob intend to engage in and which *risk* is associated with that; e.g. a purchase of an alcoholic beverage with the risk of Alice being underage, rendering the sale illegal.

[1] Some related work [1] [2] considers a slightly different scenario where Alice herself makes Claims, which are then *attested to* by an other actor, Chris.

- Which *qualities* of Alice minimize this risk to Bob or entitle her to proceed with the transaction; e.g. Alice must be at least 18 years old.
- Which *claims* Bob needs to gain confidence in these qualities; e.g. a date of birth (or simply *is over 18*) statement.
- What qualifies Chris to issue these claims; e.g. an unbiased party able to verify age, such as a notary, university or state.

2.1 Cryptographic and Semantic Layers

Status: *Done*

Existing Self-Sovereign Identity infrastructures – such as IRMA [3], Sovrin [4], uPort [5] and Trustchain [??] – realize the *cryptographic trust* in this scenario. They ensure that Alice can forward claims from Chris to Bob whilst preserving integrity and allow for selective disclosure of claim contents, or even proving something about the claim without disclosing it at all (known as Zero-Knowledge Proof of Knowledge); e.g. instead of disclosing a date of birth, simply prove that it was at least 18 years ago. The use of digital signatures ensures that claims cannot be tampered with (integrity), and the issuer cannot deny making the claim (non-repudiation). In different ways, these solutions also provide means for *revocation*; i.e. withdrawing the validity of a claim before it expires.

Alice, Bob and Chris are real world entities such as humans or their collective organisations, institutions and governments. So between a use case agnostic cryptographic layer and any *meaningful* application we must add precisely that: meaning. This calls for another layer in which actors can coordinate *what* to exchange and *whom* they trust. This layer must perform the mapping between cryptographic elements – such as pseudonyms, claims and signatures – and real world entities, social relations and business logic. We refer to this layer as the *semantic layer*.

This semantic layer could, in principle, be fully realized by applications running on a purely cryptographic infrastructure. However, we will argue in the next section why it is better to handle at least part of the semantics in the shared infrastructure.

2.2 The need for a Common Semantics Layer

Status: *Done*

As argued in the introduction, the key to a shared identity infrastructure is to find the right point of decoupling; i.e. the correct *separation of concerns*. Note that not all semantics or business logic can be fully facilitated by a shared infrastructure [6], hence applications must at least *complete* this semantic layer. Henceforth, we will distinguish between the Common Semantics Layer (CSL) and the Application Layer (AL). We will argue why such Common Semantics Layer is necessary.

2.2.1 Reuse of claims

Status: *Done*

First of all, Allen's principle of *interoperability* states that a subject should be able to reuse her claims with as many parties as may need them **allen_principles**. In other words, issuers should not need to reissue or redesign their data for every new use case or application.

2.2.2 User Control: a Commons in need of Semantics

Status: *Done*

Alice should have control over which claims are shared with Bob, and for what purpose **allen_principles**. She can obviously only exert this control through some application which properly informs her and enforces her decisions and consent. We assume that any relying applications developed by parties such as Chris or Bob do not put Alice's interests before their own. If such applications were to be given responsibility of handling (part of) Alice's control, they may be able to tamper with Alice's understanding and bias her consent, or not even respect her choice at all, without Alice being able to spot the abuse. For example, consider the popup on many European websites asking the user's consent to store cookies[2]. The content of this agreement is fully designed by the website provider, and there is no easy way for the user to be sure that her choice is actually respected: she cannot prove that she clicked *deny*.

Therefore Alice must have an application that makes transparent all that that she has and what happens to it. This application should be independent, hence free of conflicts of interest, of any other actor. We can imagine several applications being created for different user's needs, but all must be open source, transparent and deal with a multitude of use cases an a way that still provides the relevant meaning to Alice. In line with this thought, IRMA, Sovrin and uPort offer such applications called Wallets or Agents which provide a single user interface to the subject. Trustchain limits its concerns to the *cryptographic layer*.

2.3 Building on top of Trustchain

Status: *Done*

In contrast to IRMA, Sovrin and uPort, Trustchain offers full peer-to-peer anonymity by using the The Onion Router (TOR) protocol for communication. Furthermore, it uses an alternative distributed ledger technology that does not rely on global consensus, making it a scalable infrastructure. It has already been used in trials with the Dutch government and offers security at the level of the Dutch passport. This makes it perfect for studying in the context of this thesis.

As opposed to the commercial and academic alternatives, Trustchain does not yet have a common semantic layer in place. Several implementations of Wallets and other applications have been made by students, but these do not yet provide

[2] EU directive `https://eur-lex.europa.eu/legal-content/NL/TXT/?uri=LEGISSUM:124120`, Dutch law `https://wetten.overheid.nl/BWBR0009950/2020-03-01`

the desired semantic functionality. This section dives deeper into the technology of Trustchain. Section 2.4 describes in more detail the desired properties of the semantic layer.

Stokkink and Pouwelse presented a claim model for Blockchain-Based Self-Sovereign Identity [2] that meets passport-level use requirements by facilitating legally valid signatures. Their model consists of five claim metadata fields: name, timestamp, validity term, proof format and proof link. They also present three models for using claims, to satisfy different requirements: a passive, an intent-based and an active model. In the passive model the issuer and subject together sign an attestation. Any verifier can simply look up the claim metadata and verify the accompanying signatures to check its validity. However, if revocation is required, the subject may have the ability to withhold the revocation information to a verifier. The intent-based model makes this form of identity fraud evident by adding an intent block to the chain pointing to the attestation that was verified. Any auditor may now verify that no revocation was done before the verification. Identity fraud may now become evident, but it is still possible. The third model basically requests a new attestation from the issuer using a unique challenge to prevent a replay attack. In this way, revocation is basically facilitated by actively refreshing the credential. Note that this does not include any schemes.

2.4 Common Semantic Layer Properties

Status: *Done*

This section breaks down the concerns of the Common Semantic Layer.

2.4.1 Meaning and Value of Claims

Status: *Done*

The meaning of *trust* depends on the context, or in this infrastructure: on the use case. We never have *generic trust* in a party: we may trust a bank with our money but not with our darkest secrets. Hence, when we speak of trust, we actually mean *confidence* or a *belief* in a certain aspect of a party: its name, its honesty, its competence, its authority, etc. It is therefore essential that issuers, verifiers but also subjects agree on *what* is being claimed or attested to; i.e. the Meaning of the claim.

Secondly, they must understand the Value of the claim. We can think of this value as the risk in relying upon that claim; the probability of it being false and the magnitude of the consequences. As the goal of the initial exercise was to minimize the risk, we must understand this risk. We consider the following aspects:

Issuer Accountability. The Verifier may mitigate this risk by deferring it to the Issuer, either through warranty, liability or compensation. Since this puts the Issuer at risk, he must have incentive to issue the claim anyway.

Issuer Qualification. If accountability is not applicable, or insufficient for

the Verifier, the risk of incorrect claims may be reduced by understanding how the claim came to be. This problem is similar to our original problem, *Subject Qualification*, only now we have the additional constraint that evidence must be passed through the original *Subject*. The type of qualification again strongly depends on the use case; it may be honesty or competence in many forms.

ACTUALITY. When claims are based on values that can change (such as a home address or salary), an important aspect of the claim's value is its ACTUALITY. Note that the Verifier may wish to have the most current information about the Subject, so it may be risky to act on an outdated claim. We must also consider historical assertions, i.e. what was the Subject's income at the start of this year.

Standardization of meaning and value of information has been common practice for a long time. The latest development in respect to Self-Sovereign Identity is the new Recommendation from the World Wide Web Consortium (W3C): Verifiable Credentials [7].

2.4.2 Extending Trust

Status: *Done*

The value of claims is established by the issuer. If this process is somehow corrupted, the Verifier relying on that claim is at risk. The Verifiable Credentials scheme assumes that the Verifier trusts the Issuer directly to deliver correct claims. However, in many practical scenarios, this assumption of direct trust is too restrictive. Consider the following problem:

Alice is an employee of a company called Dave's. She wishes to sign a contract with a supplier Bob, in name of Dave's. Bob needs confidence that Alice is authorized to sign such contracts, so he wishes to Verify her authority. Alice provides claims signed by her boss Chris, the owner of Dave's. Bob does not know Chris, so how can he trust his claims?

The rest of this thesis will focus on problems of this kind, described in more detail in the next chapter. There are several ways for Bob to know that Chris owns Dave's and hence has the required authority. For example, a public record could list Chris as the owner. However, this approach sacrifices privacy for security. So we add the following constraint: any intermediate untrusted issuers should be able to remain anonymous so long as identification is not necessary for the use case.

2.4.3 User Convenience

Status: *Done*

The adoption of Self-Sovereign Identity will offer more and cheaper assurance in identity transactions to relying parties, and more control to subjects. However, with this control comes the responsibility to manage the collection and distribution of all these claims. This likely becomes too cumbersome for Alice, which may slow down the adoption of SSI or force an overwhelmed Alice to make poor choices that could harm her privacy.

It is often Alice's responsibility to deliver and hence fetch the necessary claims. Especially in a multi-issuer use case such as with diploma's, only Alice knows her issuer, i.e. her school. The most common approach seen in many recent SSI use cases to date is what we call *portal based*. The issuer provides a web portal where Alice can log in, using SSI claims or other authentication mechanisms. There, the relevant claims can be requested by scanning some QR code.

Many approaches seem to assume that the bulk of Alice's claims are gathered beforehand, that is before they are requested by a Verifier. Whilst we feel this is the safest approach for Alice, as she can then protect her claims for later use, we imagine the common individual will not go through the trouble of building their identity unless necessary. In other words, we believe that most issuings will happen *ad-hoc*, on a request-basis. In that scenario, the portal based approach is not the most efficient as it requires Alice to do a lot of work. It also does not offer her a consistent user experience (see Table 1.1, Cameron's Seventh Law of Identity).

2.4.4 App/System Integration

Status: *Done*

As argued above, the user needs to exert manual control over its identity operations through some independent Graphical User Interface. We assume this will primarily be operated on smart phones. In order to integrate the identity infrastructure with automated systems, such as web applications, we must also support control by a headless programmed controller. To make this semantic layer suitable for both cases, we assume the following model as our technical context:

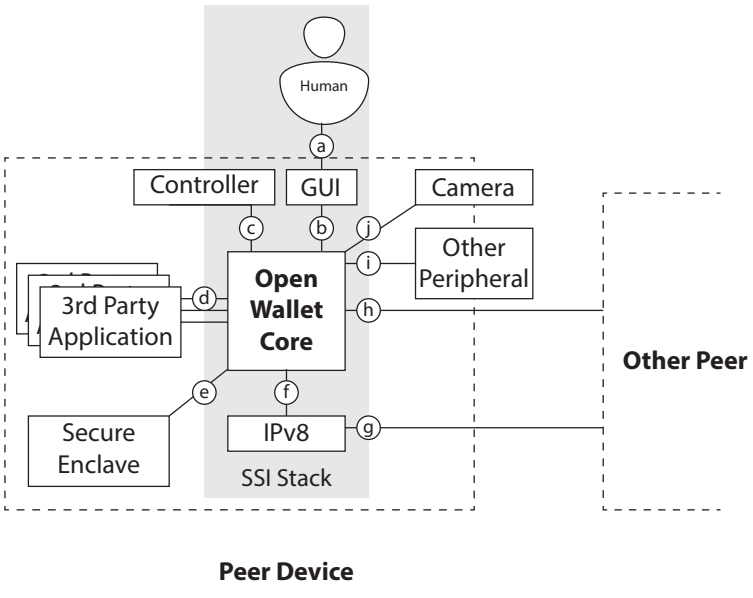


Figure 2.2: Technical Context of the Semantic Layer

Figure 2.2 shows the technical context in which the Agent operates. The subject that owns the Agent can either control it manually via a Graphical User Interface (a), or in an automated fashion using software (c).

3rd party applications running on the device may interact with the user's sovereign identity through the Open Wallet Core (d). These differ from the programmed controller (c) in terms of authority: the Controller is assumed be a trusted representative of the Subject and hence has full control, whereas 3rd party applications who need a programmable interface are not immediately trusted so their control is limited.

The Open Wallet Core assumes availability of a Secure Enclave (e) for storing keys and private data and on mobile devices it makes use of a camera (j) for scanning QR codes or other peripherals (i) such as Near-Field Communication (NFC).

It makes use of the Trustchain library as introduced before (f). The stack is designed in such a way that this library, responsible for the core identity operations such as signing, is only operated by the Open Wallet Core. Finally, each Peer can communicate with other peers running this stack through the low level Trustchain protocol or a higher level Open Wallet protocol.

3 | Case Study: Power of Attorney

Status: *Done*

The wide variety of use cases that could fit the model in Figure 2.1 makes it difficult to design for. Hence, within this thesis we will focus on a subset of problems, whilst tackling one of the more complex issues in identity systems: *delegation*. We consider the following problem:

> **Problem:** verify that a person authorized to act on behalf of some organisation.

In collaboration with the Dutch Chamber of Commerce, we develop an application that provides high legal assurance, but also convenience, in verifying such authority. This is a complex case that heavily depends on the ability for individuals to digitally identify and sign, making it an appropriate test for our system. We will limit our scope to Dutch legal entities, natural persons and legislation.

Our goal is to answer the following question in an automated fashion:

> **Assertion:** $Q(P, L, A)$ = person P is authorized to perform action A on behalf of legal entity L.

In general, i.e. for any combination of P, L and A in any legal system, answering this is extremely hard, if not impossible. Hence, for high risk transactions, a notary is called in to perform various checks, thereby consulting several registers. He then produces a *legal opinion*, which is an official statement of his findings.

For many day-to-day activities, however, such rigor is unnecessary and business can be conducted by mutual trust and the laws of *Power of Attorney* (volmacht). In this section, we derive a simplified reasoning model based on Dutch legal texts and expert interviews. We will limit our scope to those actions A for which the question $Q(P, L, A)$ can be safely answered with this model. We consider the following elements:

1. Registered directors (*functionarissen*) who control a legal entity.
2. Registered Power of Attorney (*volmachten*) granting a natural person full or partial power over a legal entity.
3. Unregistered Power of Attorney based on a shared platform.

3.1 Executive Power over Legal Entities

Status: *Done*

Depending on its form, a legal entity is owned and controlled by one or more natural or legal persons, called *directors*. This relationship is registered in the

trade register (or *Handelsregister*), managed by the Chamber of Commerce.

Human Director. The role of director may be assumed by another legal entity, which can repeat itself several times. At the top of such a chain, one or more natural persons are always in control.

Signing capacity. If a legal entity has more than one director, it may be the case that directors cannot individually sign contracts but instead must do so together[1]. We will however constrain our problem to the case where a director has full signing capacity over a legal entity.

The natural person forms the root of a tree of legal entities. The laws, and possibly other conditions, determine the extent of power this person has over each of the legal entities in the tree. We will abstract over these complexities by assuming the following:

> **Assumption 3.1.** For each combination of legal entity L and natural person P, the Chamber of Commerce has the ability and authority to state whether P has full control over L.

Next, we consider how power can be extended to other legal and natural persons.

3.2 Extending Power over Legal Entities

Status: *Done*

Whereas the human directors act as the root of the command hierarchy, enjoying unlimited control[2], they may extend their power to other legal or natural persons by means of *Power of Attorney* (or *volmacht*). The Dutch law states that Power of Attorney can be made *explicitly* or *implicitly*.[3].

3.2.1 Explicit Power of Attorney

The Chamber of Commerce explicitly registers Power of Attorney using a form (*Formulier 13*[4], see appendix). It allows to either grant the subject full authority, or restricted by options shown in Table 3.1. This also allows to fill in a restriction in natural language. As this is inconvenient for automation purposes, the Chamber of Commerce is developing a semantic model that can replace this free-form text.

The Explicit Power of Attorney method has two processes:

1. **Issuing.** To issue a Power of Attorney, the *grantor* (or an authorized representative) must fill out the form and visit one of the offices of the Chamber of Commerce together with the person being granted.
2. **Verifying.** To verify an explicit Power of Attorney, one can retrieve an excerpt (or *uittreksel*) at the Chamber of Commerce website[5] at the cost

[1] https://www.kvk.nl/advies-en-informatie/fraude/tekenbevoegdheid-per-rechtsvorm/
[2] As by our assumption
[3] Burgelijk Wetboek 3 Artikel 61:1
[4] Formulier 13 Inschrijving Gevolmachtigde
[5] https://www.kvk.nl/producten-bestellen/bedrijfsproducten-bestellen/uittreksels/

of € 2,30. The verifier must then compare the full name and date of birth of the attorney, as stated on the excerpt, with some form of legal identification.

Restriction Option	Description
By Financial Amount	Maximum amount in Euros
By Act	One or more of the following: *Requesting changes in the Trade Register, Issuing Quotations, Access RDW license plate services.*
By Contract Type	One or more of the following: *Purchase, Sales, Warranty, Lease (Rental), Financing, Software, Maintenance* and/or *a custom description*
By Establishment	Entire legal entity, or specified to a specific establishment (by address).

Table 3.1: Restriction options for registered Power of Attorney

3.2.2 Implicit Power of Attorney

Status: *Done*

The Power of Attorney method is a tedious process both when filing it and when checking it. Suppose a customer checks out at the local grocery store. Before handing over the money, he must check if the cashier actually has Power of Attorney to receive that money. As this situation is far from practical, Dutch law provides the concept of *implicit Power of Attorney*:

> Is een rechtshandeling in naam van een ander verricht, dan kan tegen de wederpartij, indien zij op grond van een verklaring of gedraging van die ander heeft aangenomen en onder de gegeven omstandigheden redelijkerwijze mocht aannemen dat een toereikende volmacht was verleend, op de onjuistheid van deze veronderstelling geen beroep worden gedaan. - Burgerlijk Wetboek 3 Artikel 61:2[6]

Freely translated, this means that if a counter party may reasonably assume that power of attorney was granted to a particular person, he may act upon that assumption. The assumption must be based on a *statement* or *behaviour* by the *grantor* and take *circumstances* into account.

In the cashier example, the customer (being the counter party) may reasonably assume that the cashier (the representative, or attorney) has Power of Attorney to receive the cash in name of the grocery store (the grantor). This can be based on the circumstances, i.e. the cashier wearing company clothing and sitting behind the cash register, and the behaviour of the grantor by (its other personnel) not removing that person.

3.2.3 Sharing Authority Information

Status: *Done*

[6] https://wetten.overheid.nl/BWBR0005291/2020-01-01/#Boek3_Titeldeel3_Artikel61_Lid2

The aforementioned methods of explicit and implicit Power of Attorney does not suit all transactions. Whereas the physical space allows customers to easily (yet superficially) assess the identity (face) and attributes (company clothing and placement) to support a *reasonable assumption*, the digital space is by default much less transparent.

This has lead to alternative approaches, such as the Dutch E-Herkenning system[7], a cross-organisational identity and access management platform. The solution roughly works like this:

1. Digital service providers to list their online products and services in a central catalog, along with the *level of assurance* they require.
2. Consumers register an E-Herkenning account for each participating employee, at an annual cost of around 5 to 30 euros, depending on the level of assurance.
3. The directors of the consuming legal entity are given full access and must subsequently grant access to specific products and services to individuals, as they see fit. They may also grant other individuals the authority to manage access.
4. As an individual wishes to use a service in name of a legal entity, they must log in with their E-Herkenning account at the right level of assurance. The account must be authorized to use the service in question.

A major drawback of this approach is that the catalog of products and services quickly becomes to large to comprehend. It also requires managers to foresee exactly which products and services their people will need in the future. When access is too restricted, employees are stalled in their work and have to bother their superiors for granting extra access. This quickly leads to dangerous practices such as lending the credentials of a colleague, or simply granting individuals with full access. ▷ Verify these issues

The aim of this design project is to provide a simple, yet effective way for employees to share authorizations when it suits them. Note that the E-Herkenning system separates the right to access services from the right to grant access to those services. In this design iteration however, we will simplify the problem by choosing these to be equal.

3.3 Actions versus Jurisdictions

Status: *Done*

In the previous two sections, we have outlined the basic forms of control natural persons can have over legal entities. The extent of this control is usually described in general terms (e.g. *sales*, or *purchases up to 5mln*). We will refer to this as the *jurisdiction* of a person.

[7] https://www.eherkenning.nl/

Before we can answer the question $Q(P, L, A)$ which considers a specific action A, we must find a way to map actions to jurisdictions. The E-Herkenning system solves this by simply expressing jurisdictions in terms of actions, i.e. the mapping is one-to-one. This solution is however not practical for many other cases, as it would require ticking a box for each possible allowed product or service.

For our current model we define a jurisdiction J as a subset of the set of all actions. We say that if and only if $A \in J$, then the action A is allowed under the jurisdiction J.

3.4 Qualification Logic

Status: *Draft*

From the previous sections we can derive the following predicates:

Predicate	**Meaning**	**Trusted Issuer**
$Full(P, L)$	states a person P has full control over a legal entity L.	Chamber of Commerce
$PoA(P, L, J)$	states that a person P has registered Power of Attorney over L restricted to some jurisdiction J.	Chamber of Commerce
$Auth(P, L, J)$	states that a person P is authorized to act in name of L within the boundaries of some jurisdiction J.	Any qualified person

Table 3.2: Predicates for determining authority

Note that we cannot assume truth, so we depend on claims by various parties. We mark the issuer of a claim by a subscript; $Full(P, L)_{KVK}$ means that the Chamber of Commerce (KVK) states that P has full control over L.

$$Full(P, L)_{KVK} \rightarrow Q(P, L, A) \quad (3.1)$$

$$PoA(P, L, J)_{KVK} \textbf{ and } A \in J \rightarrow Q(P, L, A) \quad (3.2)$$

$$Auth(P, L, J)_P \textbf{ and } A \in J \textbf{ and } Q(P', L, J') \textbf{ and } J \leq J' \rightarrow Q(P, L, A) \quad (3.3)$$

▷ We can also simplify by using $Q(P, L, J)$

▷ $Q(P, L, J) and J \geq J' \rightarrow Q(P, L, J')$

TODO: *Check formal logic notation*

[8]

Note that claims $Full(P, L)$ and $PoA(P, L, J)$ are only trusted when issued by the Chamber of Commerce. In contrast, a claim $Auth(P, L, J)$ can be made by

[8] https://en.wikipedia.org/wiki/Propositional_calculus https://en.wikipedia.org/wiki/First-order_logic#Deductive_s

any party P', but then requires that P' has sufficient jurisdiction; i.e. $Q(P', L, J')$ where $J' \geq J$.

As rule 3.3 is recursive, one may have a chain of claims that end with the subject P. For this chain to be valid however, it must start with a person P' for which one of the base cases hold; i.e. $Full(P', L)$ or $PoA(P', L, J)$.

Furthermore, we observe that in all cases the *grantor* reserves the right to revoke its claim.

3.5 Usage Scenarios (or Context)

Status: *Placeholder*

Answering the question $Q(P, L, A)$ is primarily relevant when parties are about to perform action A. Hence, our application must be able to provide this answer at that moment.

We first distinguish two usage scenarios: virtual and physical verification. In the virtual scenario, the subject P attempts to use an online service. Before proceeding, the service provider needs to know who P is, which legal entity L is involved and whether P is allowed to do A on behalf of L.

The physical scenario works rather similar. The counter party is now a human being asking the subject P for the same information. In this scenario we have the added benefit that the counter party can visually identify P.

In both scenarios we leave it to the counter party to map the desired action A to a jurisdiction J as defined in the system.

TODO: *Describe Scenarios. May fit better in design section.*

TODO: *Add identifiers of Legal and Human entities*

4 | A Theoretical Framework for Self-Sovereignty

Status: *Done*

This chapter presents a model for debating, analyzing, designing and evaluating *digital sovereignty* at a high level. It raises important questions and provides structure and clarity to debates in this field. This enables us to solve these complex problems. Moreover, we address issues within the current state of the field:

1. It helps close the gap between governance and practical execution.
2. Solutions, protocols and standards are likely to be implicitly based on assumptions and values. Making these explicit may expose vulnerabilities and fundamental conflicts, but also unintentional discrimination or exclusion.
3. The field still lacks a proper method for evaluating and comparing solutions and their underlying beliefs. This is necessary to unify the different findings of practical and theoretical exercises done around the world.

The model offers a way to put smaller efforts into the larger perspective and expose possible conflicts early on, such as distributed ledgers possibly harming an uncompromising need for privacy.

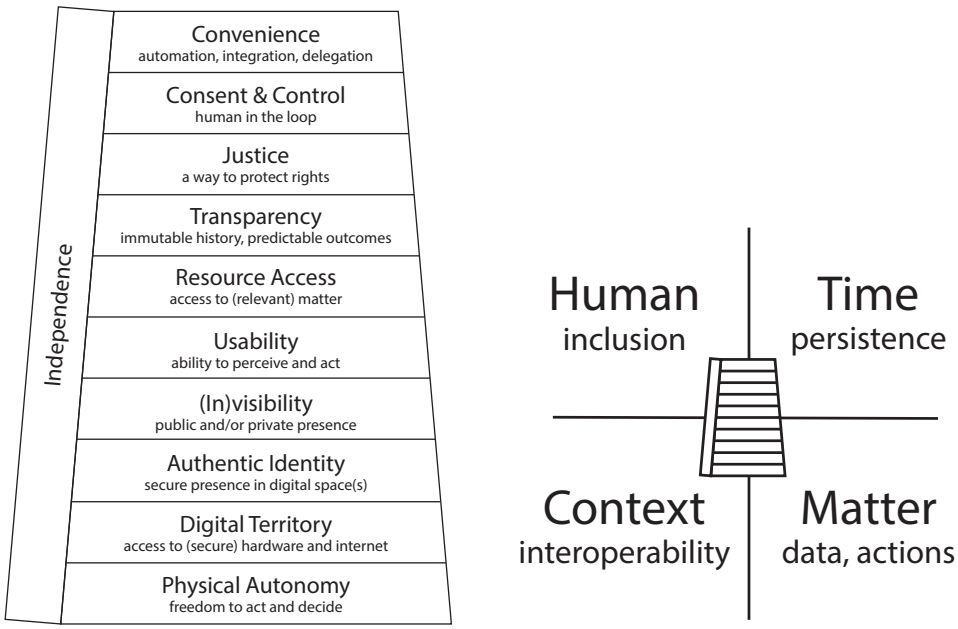
The model combines insights from the work of colleagues – not only Cameron and Allen, but also theoretical and practical exercises in the field – with insights from expert interviews and lessons from the design iterations made during this project.

The model is visually represented in figure 4.1. Figure 4.1a presents 10 stacked principles that constitute the *pillar* of digital sovereignty. Each layer is a building block needed to realise digital sovereignty and depends to a certain extent on the layers below it. The 0^{th} principle, Independence, acts as the *spine*. It considers the (possibly vulnerable) dependencies upon technology, actors and other factors that uphold the pillar.

TODO: *Add the Subject of the principles*

The pillar can be applied in four ways:

1. **To Debate**: debate the desired situation for a sovereign user and translate this to requirements of a system.
2. **To Analyze**: discover edge cases and threats to the desired situation, such as rogue actors.



(a) Pillar of Digital Autonomy (b) Dimensions of Problem Space

Figure 4.1: Model for Digital Autonomy

3. **To Design**: design and build systems that meet the established norms and requirements.
4. **To Evaluate**: compare designs or existing systems with each other or with a set of established norms.

4.1 Dimensions

Status: *Done*

The complexities at each layer of the pillar can be further explored through four dimensions. Figure 4.1b shows the dimensions Human, Time, Context and Matter. As digital systems attempt to establish (in part) the autonomy of an actor, they do so in a particular context, taking into account a subset of all people, considering specific matter and assuming a particular time frame. These considerations may conflict with other subsets of the problem space.

The next section explains each principle.

4.2 Principles

Status: *Draft*

TODO: *What do principles strive for? Is it a binary quality? Is it a measurable quantity? Or do we simply have to consider a particular shape?*

4.2.1 Principle 0. Independence

In the digital world, it is impossible to be fully independent. Our online activities can only be realized through the devices we operate, the software that they run and the infrastructure that binds them together. Each step of this process involves not only technologies, but also actors such as Internet Service Providers and *trusted third parties* that enable, monitor and govern.

In establishing each level of digital sovereignty, we must be aware of such dependencies and their possible conflicts of interest, as this may lead to unwanted surveillance or discrimination, undermining our self-sovereignty.

4.2.2 Principle 1. Human Act

If an actor is not autonomous in the physical world, it cannot be autonomous in the digital. One's digital avatar may however do all kinds of things that the human behind it never dares to do in the physical space. But it is still the mundane freedom, that of the thought, that enables the person to take those actions in virtual space.

In reality, the freedom of the human is always limited, by its motivations, norms and values, and other external factors. Examples of those factors are (superior) authorities such as parents, employers, society and legislation, but also physical, financial and mental limitations.

4.2.3 Principle 2. Digital Territory

Obviously, for the human to be autonomous in the digital world, she must have access to it. This starts with access to the internet, preferably reliable, free of discrimination and surveillance. However, *access* alone is insufficient for sovereignty, as the individual needs a place to store the data that is rightfully hers and execute the logic that serves and protects her needs.

Ownership of a smart phone or other device is a key assumption that virtually every Self-Sovereign Identity project makes. It provides a space to store (part of) cryptographic keys, private data, receipts and other *evidence*. It also enables execution of logic for signing, verifying and data sharing, all on a medium that is physically in possession of the individual.

This fundamental building block is also the Achilles' heel of digital sovereignty. If the phone breaks or is lost, the individual loses all her possessions. Even worse, the accumulation of all this value may make it an attractive target for theft. Furthermore, by requiring individuals to wear a smart phone at all times, it may expose people to even more surveillance.

4.2.4 Principle 3. Authentic Existence (Identity)

In the digital world, the individual interacts with other parties and systems basically by sending messages. The ability to determine that two messages have the same sender, i.e. that senders are *identical*, is the ability to establish *identity*. Many parties are interested in this ability, or power, for very different reasons:

The individual wishes to withdraw money from her bank account. The bank

must track down a physical human if she fails to pay her debts. Advertisers build behavioural profiles for targeted advertising. Surveillance states monitor threats. Hence, from the perspective of the sovereign individual, *correlation* may be used for good and for bad.

In any case, we link all accumulated value (money, reputation, information) to an **identifier** such as a bank account number, an e-mail address or public key. This is in stark contrast to the exchange of value through cash money, which is highly anonymous.

Comparing identifiers is easy, but the difficulty lies with linking the individual to the identifier. This is why so many means of authentication exist. The extent to which this authentication provides certainty is called *level of assurance*. High value and low assurance open up the possibility of identity fraud.

Identifiers can also act as a network endpoint. Through some network and routing we can assure that given an identifier, one can contact the individual to which that identifier belongs.

Ideally, we have a means to lay claim to value without being linkable everywhere.

4.2.5 Principle 4. (In)visibility

The individual may wish for a (semi-)public appearance, such as in social networks, or on trading platforms. However, the individual must also be able to operate in private because this prevents self-censoring, or discrimination by others. Hence, a proper balance is required depending on the desired sovereignty.

4.2.6 Principle 5. Usability

If the individual does not understand what is going on, cannot reason about it, or is unable to provide the appropriate inputs, she cannot be sovereign. This covers not only the fields of accessibility and human computer interaction, but also the ability to comprehend the digital world in which she operates.

However, this should not be an argument to withdraw sovereignty from *all* individuals. Instead, we should start with maximum sovereignty and gracefully support those who are unable to use the system. This can be done by creating adaptive systems, setting defaults, but also offering support and eventually facilitate delegation to trusted people.

TODO: *Use term Competence*

4.2.7 Principle 6. Resource Access

The (meta)data over which the individual has (shared) authority, should be accessible to her and only those who are authorized. Currently, many individuals' data is stored in large data warehouses, silos that they cannot access or only *peek* into.

These *resources* should be accessible in such a way that data can move freely without losings its value. This allows the individual to move or copy it to her own *territory* for protection (similar to why one would save a purchase receipt),

and/or reuse in another situation.

4.2.8 Principle 7. Transparency

Insight in what has happened, is happening and what is about to happen provides the actor with the necessary knowledge to act autonomously. If the laws that govern the system are not transparent, the actor cannot see nor predict outcomes. Increasing transparency could hence improve trust. We must however also consider that this may conflict with privacy.

4.2.9 Principle 8. Justice (and Support?)

Money cannot be spent twice. Each context has its own rules, its own idea of justice. For an individual to operate in those contexts, it must be assured that justice is served in one way or another. This may be built into technologies such as distributed ledgers, or brought by trusted authorities. In any case, it is important to think about what *justice* is in the context of interest, how this is enforced and what happens if rules are violated.

Unifying systems and contexts on this principle is likely to be the most difficult, as it comes back to unifying conflicting ideas of justice. However, rules will always be baked into systems, and we should rather make this explicit beforehand than let the biggest company decide what justice is.

4.2.10 Principle 9. Consent and Control

To be fully sovereign, an individual must be able to exert influence over those matters that he has authority over. This starts with asking the individual for consent, but can extend to providing controls. Note that these depend on all the underlying principles.

Proper consent requires the actor to be capable (principle 5) and well-informed (principles 6 and 7) to make the decision at hand [GDPR]. Moreover, it should not be easily forged, and the consent should be enforced.

The actor can only be truly autonomous if it is in control over the relevant matter. Instead of being asked if the actor is “OK with taking a left turn”, the actor itself can now take the wheel.

4.2.11 Principle 10. Convenience

When the actor is in control, it may be burdened with a lot of effort: with great power comes great responsibility. Hence, if the actor is free to automate this, integrate it with other systems or delegate it, the burden might be relieved and full autonomy can be established.

4.3 Architectural Principles

TODO: *Process Georgy’s feedback*

4.3.1 Permission-less

There should be no gatekeeper that decide who can join the network and who cannot.

- As virtually any party operating online needs digital identity, it is very impractical and expensive to submit all actors to audits before they are allowed to enter. This will either cost us time or quality of assessment, and it makes identity an exclusive good again.
- This network must be able to operate across the worlds, in different context and different jurisdictions, each with a different set of rules. If there is overlap in these rules, it is probably very expensive and hard to find, as well as insufficient for many cases.
- The same, if not better, actor quality assessment could be accomplished in a decentralized manner. Automatically verifiable certificates of quality are no different from other identity claims.

4.3.2 Simple

Even if *unpermissioned*, the system and its possible governance framework should not discourage issuers, verifiers and users by making it utterly complex to participate. The e-IDAS regulation for identity and trust systems refers to other regulations, which in turn refer to yet more regulations. All in all, over 400 regulations apply, making it very complex to partake, leading to recentralization of identity systems. Hence, implementation and entry should be relatively simple.

4.3.3 Sovereignty First

Many jurisdictions and political systems have different views on the level of autonomy they wish to grant their citizens. Furthermore, these views are likely to change over time. An infrastructure that is meant to operate across borders and decades must be able to overcome these conflicts.

I see two driving forces oppose the individual's autonomy: **security** and **inclusiveness**. First, security may demand insight in certain activities and impose restrictions upon them. We should avoid the all-seeing eye, but make it easy for participants to report and prove possible abuse. This is comparable to the physical world that starts with autonomous humans who then create social and technical constructs to ensure security.

Second, the goal to include all humans is very important, but also imposes serious limitations upon the system. Some view humans as overall very incapable and opt for as little involvement as possible; i.e. "let the clever ones handle it". In certain cases, this may be appropriate. However, if these decisions limit the freedom of all human actors in the system, it harms their sovereignty. Several means are possible to support those who need it, whilst leaving sovereignty of others untouched. So *support instead of enforce*.

4.3.4 Avoid Consensus Lock-In

As illustrated in the principles above, decisions may be made on global level that impact all individuals and use cases. It is not only hard to acquire, but also likely to result in sub-optimal solutions and a dramatic slow down of development. Instead, we aim to exclude many of these matters from our architecture and

facilitate local consensus making, allowing smaller groups of actors to collectively discover new solutions. This means additional complexity as we must be able to cope with these different decisions.

4.3.5 Minimal Governance

Perhaps necessary to mention separately again, but the motivation has already been provided.

4.3.6 Maximize Modularity/Incrementality

In the same trend as the previous principles, because consensus is hard, we should be able to develop this network in steps and modules. This ensures that different solutions may be experimented with, and incrementally adopted. It also allows conflicts to enter the system; between policies, protocols, configurations. We must be able to cope with those. In other words, we must unify the right matters, but let the rest be free.

4.3.7 Peer Equality

We do not distinguish between different types of actors. All peers are created equal and should have the ability to perform all operations, including issuing and verification. They do not, by design, depend on special types of peers.

4.3.8 Transparent

The workings of the system, including any possible governance, should be as transparent as possible, whilst preserving the Sovereignty First Principle. As it is an infrastructure of the people, the people should have the right to observe and debate its behaviour.

5 | Cryptographic Layer

Status: *Draft*

In this chapter we analyse the infrastructure upon which we build our identity solution: the TrustChain Stack [8]. We describe its functionalities, the guarantees it provides and the input it requires. We also discuss its applicability to the identity problem at hand.

The TrustChain stack consists of:[1]

- IPv8, a peer to peer networking library [9].
- The TrustChain protocol, a personalised distributed ledger [10].
- An application layer that exposes a REST API for identity operations [11].
- A claim model for self-sovereign identity [2].

IPv8 is a network overlay in which peers, distinguished by public keys, can directly set up encrypted communication with other peers by addressing their public key. In contrast to the Internet Protocol [12], which facilitates device-to-device communication, IPv8 allows one peer to directly address another peer, regardless of the device they are operating. This is a much better fit for human to human communication, upon which our identity problem is founded.

Its modular architecture ensures that IPv8 can evolve incrementally. It decomposes functionality into different network overlays, also known as communities. These communities are simply groups of peers running the same protocol to accomplish some goal. If they wish to adopt new functionality, peers only have to join another network overlay. This creates an ecosystem that is self-regulating and self-evolving. We analyse three primary functionalities in this networking layer that are of interest to the identity problem at hand:

1. Establishing a connection between any two peers.
2. Attesting to a peer's identity claim.
3. Verifying a peer's identity attestation.

In the next sections, we discuss each of these functionalities. Section 5.4 describes the accounting mechanism that IPv8 uses for identity attestations: the TrustChain protocol. As the purpose of this layer is to create trust in identity claims, we must specifically combat *identity fraud*. This is discussed in section 5.5, and specifically information withholding in section 5.6. Furthermore, we discuss two vulnerabilities of this layer when applied to real world identity use cases: metadata leakage (in section 5.7) and re-use of identifiers (section 5.8).

[1]The TC IETF draft in this case is confusing me, as it decomposes TC stack into (1) application layer, (2) network overlay and (3) ssi layer, but also mentions the trustchain fabric and trustchain protocol. There is no reference to the claim model.

5.1 Peer to Peer Communication

To set up a connection with a peer, its network address must first be found. The IPv8 overlay *DiscoveryCommunity* executes the peer discovery protocol. All peers together operate a Distributed Hash Table that maps a peers public key to its current network address. It employs *gossiping* to spread this information and also propagate connection requests through the network to ensure that any peer can *ask around for* any other peer, with the help of other peers in the network. IPv8 also solves an additional complication, traversing Network Address Translation (NAT) by employing NAT puncturing. This enables smartphone-to-smartphone.

So, in order to set up communication, we need a peer id (a public key or a hash of it). By gossiping this request to other peers in the network, they learn that which peers are establishing a connection, which may be a risk to privacy, especially if the real identity of those peers is known.

5.2 Attesting to a Claim

IPv8 distinguishes between *attributes* and *attestations*: a peer *claims* that some statement about his attributes is true. Other peers may provide an *attestation* to that claim, thereby *vouching* for its truthfulness. The relying party, also known as the *Verifier*, can use these attestations to gain confidence in the truthfulness, or *veracity*, of the claim.

This model supports both *authoritative qualification*, as well as *peer qualification* or a combination thereof. In authoritative qualification only the attestations of particular peers are trusted, such as Certificate Authorities in X.509 Public Key Infrastructures [13]. In contrast, peer qualification networks such as the PGP Web of Trust model [14] allow any peer to attest to an attribute. Trust is computed based on certain metrics such as number of attestations. This thesis focuses on authoritative qualification, because it is present in many of the modern day identity use cases.

Field	Description
Name	The name of the attribute
Timestamp	The time of claim creation
Validity term	The time after which the claim is no longer valid
Proof format	The type of proof for the claim
Proof link	The strong link to the proof for the claim

Figure 5.1: Claim Metadata[2]

The IPv8 overlay AttestationCommunity provides the protocol for creating and verifying attestations. IPv8 considers attestations a transaction between two parties: the attester, or issuer, and the attestee, or subject. Both parties sign the transaction, which ensures that no identity operation will occur outside the control (and consent) of the subject, but also not without control of the attesting peer. TrustChain explicitly excludes mono-signature support [8]: “Trustchain is based on the assumption that both parties agree on the transaction before

signing it, making tampering inherently easy to detect." These signatures offer three qualities: integrity, authenticity and non-repudiation.

Figure 5.1 shows the data that is included in the attestation. As with any transaction, attestation blocks are considered public in IPv8. Section 5.7 will discuss the possible privacy implications of this design choice.

Transactions are captured in a personalized block-chain-like data structure, called TrustChain. Each transaction is captured in a *block* that contains the content of the transaction, the two signatures and also a reference to the preceding block of each party. This creates a personal chain that is entangled with other personal chains. Section 5.4 describes this in more detail

5.3 Verifying an Attestation

The Verifier may obtain the attestation by requesting it, or through gossiping or crawling of blocks that occurs in TrustChain. The Verifier can check the signature of the transaction to ensure that the attestation was not tampered with and was made by a trusted peer. He can check that the remaining metadata such as timestamp, expiration and attribute name meet his requirements. Note that this can be done without the subject's knowing, let alone control.

The content of the attestation can be anything, such as the plain value of an attribute. However, for any attribute that is considered personal data, this should not be publicly available. For this reason, IPv8 has built-in support for interactive Zero-Knowledge Proofs (ZKPs).

5.3.1 Zero-Knowledge Proofs

In general a Zero-Knowledge Proof is a method whereby one party, the *prover*, can prove to another party, the *verifier*, that he knows some value without revealing that value[2]. In an interactive Zero-Knowledge Proof, the verifier sends a number of challenges to the prover. Only if the prover knows the particular value he claims to know, he can win all challenges and send back a response confirming that.

Instead of a plain attribute value, a ZKP *statement* is provided in the attestation. This statement contains the assertion that the subject *knows* the value, but it does not contain the value itself. Because no private data can be derived from the attestation itself, it can safely be shared with the verifier. As the statement was created together with the *attester*, the verifier can rely on the value to be truthful. Several types of Zero-Knowledge Proofs are possible, the simplest of which is *equality* [15]; the proof can only succeed if the attribute has a specific given value (e.g. *city* equals *Delft*). In contrast, a *range proof* only succeeds if the attribute value lies within a particular range (e.g. *age* is at least *18*) [16] and a *set-membership proof* proves that the attribute value is a member of some set (e.g. *country* in *list of European countries*) [17].

[2] `https://en.wikipedia.org/wiki/Zero-knowledge_proof`

5.3.2 Sharing knowledge obtained in Zero-Knowledge

A successful proof convinces the Verifier that some attribute meets certain constraints (equality, inequality, set-membership, etc.) assured by the attester. However, due to the *Zero-Knowledge* property of Zero-Knowledge Proofs, the Verifier cannot trivially convince any other party of this fact. The intuition is that any party can generate a transcript of a zero-knowledge proof (a series of challenges and responses) that is indistinguishable from a real transcript. Hence, the Verifier cannot convince a third party that the subject successfully proved something. This also makes it impossible for the Verifier to prove to an auditor that it properly verified someone's identity, which may be required by Know Your Customer and Anti-Money Laundering regulations.

In practice however, peers may still trust the verifying party with being honest about the interaction. This is both an opportunity and a threat. The threat being the leakage of private data. For example, to a company sharing private data with advertisement companies, there is no need to actually *prove* the data is correct. It has incentive to be honest about this data because lying could harm the business relationship with the advertiser. We should therefore not let the use of Zero-Knowledge proofs provide a false sense of privacy, and still minimize the sharing (or *proving*) of private data.

Existing trust in a verifier can also be considered an opportunity, in fact minimizing leakage of private data by introducing a trusted middleman. By verifying one or more of the subject's attributes in zero-knowledge, the verifier can *validate* that the subject meets some set of criteria. For example, he can verify the attestation from a university that he is a student. He can then issue an *is_student* attestation. Because this attestation comes from a third party, it cannot be directly linked to the university, so the subject gains privacy.

5.4 TrustChain, a personalized ledger

As mentioned in section 5.2, attestations are considered transactions, which are all recorded in a distributed ledger. Specifically, IPv8 relies on a personalized ledger called TrustChain. In other common block-chains, such as Bitcoin **bitcoin** and Ethereum **ethereum**, peers share a single ledger. Peers hold a full or partial copy of the ledger and over 50% of peers must accept a transaction before it will be appended to the ledger. This is known as global consensus. This feature has dramatic implications for the ledger's speed. Bitcoin's theoretical limit is seven transactions per second.

TrustChain lacks global consensus, which provides it with superior horizontal scalability. As a consequence, transactions on TrustChain represent local consensus between two actors at a particular time. Figure 5.2 shows TrustChain. Because each block holds a pointer to the previous block of both parties, their chains become entangled. In principle, blocks are only stored by the peers that were involved in those transactions. This maximizes horizontal scalability in terms of storage. However, this also allows peers to withhold transactions. This is known as the information-withholding attack or, from cryptocurrency jargon,

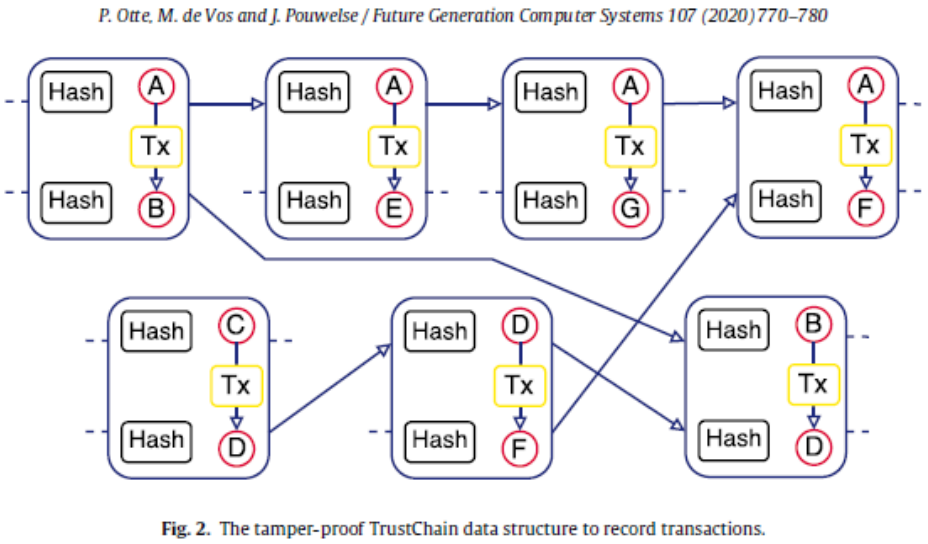


Fig. 2. The tamper-proof TrustChain data structure to record transactions.

Figure 5.2: TrustChain[10]

double spending attack, since hiding a payment could allow a peer to spend that money again. By design, TrustChain is not tamper-proof but tamper-evident. The peer can withhold information, but as the same information is also present at another peer, this may show up later and provide evidence of tampering.

Like many block-chains, TrustChain was developed with economic transactions in mind. It originated from a file sharing project called Tribler. For economic transactions of fungible goods such as bandwidth, money or other economic assets, this can work well. The question is whether TrustChain can be applied to identity information in the same way.

5.5 Identity Fraud

IPv8 uses the TrustChain to "provide the transparency and persistence needed for audit logs and the legal status of the identity system." [2]. Its goal is to expose possible identity fraud. In this section we examine possible cases of identity fraud in our Alice, Bob and Chris scenario and discuss how each of these frauds can be combated.

We consider identity fraud in the following broad sense: Alice tries to convince Bob of a *truth* that is not, or no longer, true from Bob's perspective. Note that we consider *truthfulness* a subjective property; in the eyes of Bob. Bob relies upon Chris to determine truthfulness of Alice's claim. We distinguish the following scenarios:

1. Alice convinces Chris to attest something that is not true.
2. Alice forges an attestation by Chris without his involvement.
3. Alice manipulates an attestation by Chris to a different meaning.

4. Alice uses an existing attestation that does not belong to her.
5. Alice hides an attestation, pretending to Bob it does not exist.

We discuss each of these attacks briefly:

Issuer Scamming. In the first attack, Alice may fool Chris into thinking that she is someone else, or has certain qualifications (e.g. by cheating on an exam). It is up to Chris to establish a level of certainty in the truthfulness of the claim, before making an attestation. Ideally, he communicates his level of certainty to Bob, so the risks of relying upon the attestation are known. This is done in governance or trust frameworks such as the European regulation *e-IDAS* for *Electronic IDentification Authentication and trust Services* [18] and *Digital Identity Guidelines* by the National Institute of Standards and Technology [19], which we will discuss in more depth in the next chapter.

Attestation Forging. The second attack is mitigated by the use of digital signatures, particularly the authenticity property. In order to forge an attestation, Alice must either break the encryption or steal the private key of Chris. It is up to Chris to protect his private key.

Attestation Tampering. The third attack is also mitigated by use of digital signatures, particularly their integrity property. As the transaction including metadata is signed by Chris, the content of the transaction cannot be changed without Bob being able to notice this. So again, Alice must either break the encryption or steal the private key of Chris.

Identity Binding. The fourth attack relates to identity binding. Suppose Chris provided another peer Dave with a valid attestation, the attack is such that Alice convinces Bob that this attestation applies to her. By design each identity transaction is bound to its subject by mention of its public key. If Alice could somehow obtain Dave's private key, she can pretend to be Dave, committing identity theft. Dave could also share his identity with her on purpose; consensual impersonation or identity sharing. Finally, Alice could perform a man-in-the-middle attack, convincing Dave to execute the proof to Alice whilst Alice forwards his messages to Bob. We address these attacks in the semantic layer.

Information Withholding. We discuss the fifth attack in more detail in the next section.

5.6 Information Withholding

The final attack relates to information withholding (also known as double spending). Recall that by the principles of self-sovereign identity, Alice is in *control* of her data and hence decides whether she shares information or not [20]. In that sense, information withholding would be intentional functionality; a feature, not a bug. This limits the applicability of self-sovereign identity to all those transactions over which Alice has authority. Hence we specified the attack to "pretending to Bob it does not exist". In general, Alice would hide

information when that is beneficial to her, i.e. *negative information*.

5.6.1 Absence of information

A common practical use case of negative information is that of a criminal record. Employers wish to know whether a new applicant has not committed any (relevant) crimes. The applicant is the authority in this information sharing scenario, yet how can he prove the absence of such information? In the Netherlands, the ministry of Justice and Security provides citizens with a *certificate of good behaviour* ("verklaring omtrent het gedrag") which shows that the state is unaware of any relevant criminal records of the subject. So, one approach is to trust a third party to observe the absence of negative information.

5.6.2 Updates and Revocations

Another likely scenario is when an attribute value changes. Attestations issued on previous values of the attribute should no longer be considered valid. For example, the subject may *move* any time, rendering the address attribute invalid. In some cases, the actuality of information is essential, such as when an employee is fired but can still use his credentials to sign contracts or take other actions that could harm the employer. By the SSI principle of control, the attester cannot force the subject to agree to the new information. Even if the subject *did* sign the information, he can still withhold this from a verifier. Also, the SSI philosophy does not allow the verifier to directly contact the attester, or vice versa, to check if the information is still valid.

Stokkink and Pouwelse offer two ways to support revocation on personalized ledgers such as TrustChain. They distinguish two models: *intent-based* and *active*. The intent-based model records the verification intent as a new transaction on the TrustChain. Suppose the used attestation was made at time T_i, and verified at time T_v, the verifier and subject sign a transaction pointing to the original attestation. If at a later audit it turns out there was a revocation block between T_i and T_v, then this shows the subject has withheld relevant information, hence committed identity fraud.

Because certain use cases need immediate certainty, they cannot rely on later audits to reveal fraud. This problem is tackled by the *active* model, in which the subject simply asks the attester for an updated attestation. It uses a unique challenge given by the Verifier to ensure that the subject cannot replay the signature from an earlier attestation. Note that the attester must be online during this transaction.

5.7 Meta-data Leakage

Transactions in TrustChain are public by design, not only for auditability but also for availability. Peers may hold copies of other blocks they are not involved in to improve the availability of data in case peers go offline. This makes sense in case of file sharing communities which use transaction history to allocate bandwidth, i.e. don't share files with peers that only leech instead of seed. It would be infeasible to require all peers to be online for such algorithms to operate

properly. Yet in the case of digital identity, this argument for availability only holds for public attributes that should be verifiable without involvement of the subject. The primary use case of self-sovereign identity lies in those presentations in which the identity owner actively controls and consents to the sharing, and ensures minimal disclosure to only specific parties.

Does the public nature of attestation metadata conflict with the privacy constraints imposed on our identity problem? Table 5.1 shows an example set of transactions that belong to the same subject. This includes the metadata as explained in Figure 5.1 and the public key of the issuer as shown in Figure 5.2, and also includes revocation blocks and intent-based blocks (as explained in section 5.6.2). We included another column that mentions the real-world identity of the issuer. Note that we excluded the attestation value, or *proof link*, as it offers no meaningful information here.

Timestamp	**Attribute Name**	**Issuer**
Aug 1, 2010	eligible_to_drive	KRs42.. (Centraal Bureau Rijvaardigheid)
Aug 15, 2010	is_resident	XdaST.. (Delft, Netherlands)
Sep 1, 2010	is_student	AT1da.. (Delft University of Technology)
Sep 12, 2010	has_bank_account	12opT.. (ING Bank)
Dec 31, 2010	insured	IEQW9.. (Insurance4Students)
March 13, 2013	owns_company	YY21x.. (Kamer van Koophandel)
Aug 31, 2013	has_diploma	AT1da.. (Delft University of Technology)
Nov 5, 2013	is_employee	RQ231.. (Creative Media Company)
Dec 31, 2014	insured	pRI12.. (Healthy Insurance)
Feb 9, 2016	is_employee	hK55x.. (Trucking Company)
Jul 7, 2020	has_diploma	AT1da..(Delft University of Technology)
March 1, 2024	is_father_of	RYsW3.. (Rotterdam, Netherlands)

Table 5.1: Example of public metadata of single peer

Transactions do not include the real world identity of the issuer by default, only its public key. However, for verifiers to be able to trust that the attestation came from a particular source, the (authoritative) source they trust, the link between real world identity and public key must be publicly known. Hence, we may assume that in the current design of IPv8, all metadata listed in Table 5.1 is public. Whilst the values of the attributes remain hidden, it still reads as a pretty detailed biography. Even without knowing the real world identities of issuers, sensitive information would be revealed. This is a major privacy concern.

5.7.1 Metadata Confidentiality

However, attestation metadata cannot be fully considered private either. The Verifier requires the metadata of the attestation he is to verify. Without it he cannot understand it, nor ensure he trusts the attester. Once this information is disclosed to the Verifier, the system has no way to prevent the Verifier from spreading this information. From this perspective, the creators of IPv8 reason that metadata should not contain any sensitive information and hence be

considered public.

As mentioned before, for many meaningful identity applications, the metadata inevitably contains sensitive information. Hence, we must expand the security model to include the Verifier. We must trust, or enforce through external measures, that the Verifier will not forward this information to other parties without consent of the subject. Modern data protection regulations already restrict the Verifier's capability to share information. The risk of penalties motivates issuers to comply. If we can establish that the Verifier is susceptible to these external forces, we can consider the attestation information as confidential and limit the sharing of that information only to the verifiers that need it.

5.8 Re-use of Identifiers

TBD: when peers reuse the same identifier in different transactions and contexts, this may have serious impact on privacy.

6 | Semantic Layer

Status: *Placeholder*

6.1 Semantic Security

Status: *Placeholder*

The primary security aspect of the semantic layer is that of the binding between a claim and an entity (non-transferability, identity fraud), as opposed to claim-pseudonym binding in the previous chapter. But we could consider more security threats such as over-sharing, information withholding,

6.2 Nym-to-Entity

Status: *Bullet draft*

In this section we describe how we distribute pseudonyms over entities and how their ownership/control means impersonation.

- The actors in our problem scenario – Alice, Bob and Chris – are considered to be real world entities such as humans or groups of humans such as organisations.
- To utilize the cryptographic layer, we must map these real world entities to pseudonyms.
- We model our system after the public key cryptography paradigm (source?):
  - A pseudonym *impersonates* such a real world entity. However, to prevent correlation, we argue that entities should be allowed to use more than one pseudonym.
  - Each entity creates its own pseudonym with associated keys, to ensure that no other party knows this secret. Otherwise, said party could impersonate the subject.
  - We assign responsibilities of pseudonym management and protection to the entity that it impersonates.
  - Note however that pseudonyms or the devices that carry them may be stolen or copied, which means that a binding between an entity and a pseudonym cannot be assumed persistent.
- Optional: Note that representation/delegation differs from impersonation. The former recognizes the distinction between the subject and its representative and requires approval/authorization. The latter makes no distinction between entities, as they are believed to be the same. By extension, we assume in impersonation that the actions made by the

pseudonym are made by the entity it impersonates. In representation, the representative may act differently than the subject would have.

- As with the public key paradigm, we have the problem of proving the binding between a real world entity, or one of its identities and its pseudonym.
- We apply the same claim-based identity paradigm to this problem, which means a new instance of the problem scenario: Chris validates that Alice owns a pseudonym A, and issues a claim that binds a known identifier of Alice (say her full name) to her pseudonym. If Bob trusts Chris to be honest and competent about validating Alice's identity, he can use this claim to know that Alice owned pseudonym A at time t_i. Bob must also know Alice by the same identifier.
- If Bob needs rely on this binding at a later time, we have two options. Either Bob trusts Alice to protect her pseudonym and issue a revocation as soon as necessary, or Chris needs to re-validate the binding. One option is for Alice to share a secret (PIN code or password) with Chris which she reveals to Chris at verification time, allowing Chris to issue a new claim. Another method is described in the IDEMIA case study (see X).
- How can Chris validate that Alice owns A? He can send a challenge to one of her known identities over a trusted channel, which she can sign with her pseudonym. Assuming there is no reason why an actor other than Alice would want to answer this challenge, we assume that the pseudonym must belong to Alice. (what if Alice stole A?)
- We assume a method exists that allows the identity-to-pseudonym binding as described above.
- Note that for many cases we do not need to identify Alice. All we want to know is that given attributes belong to A, for which we must know that the entity that the issuer thought controlled the pseudonym at that time must be the same entity that controls the pseudonym during verification.
- In the above model, Chris is the Certification Authority that validates the binding. This is exactly like the X.509 model. *To what extent should I describe this?*

6.2.1 Qualifications

Status: *Placeholder*

- Bob wishes to know that Alice is qualified for their intended transaction, without necessarily knowing Alice.
- As we have defined Alice to be impersonated by a pseudonym A, Bob can trust a qualification of Alice if and only if he can trust that:
  - Alice controls pseudonym A at the time of verification.
  - A qualifying claim was issued to A at the time of issuing by Chris.

  - Chris validated that Alice controlled pseudonym A at the time of issuing.
  - Chris is competent and honest in asserting Alice's qualification.
  - Chris is competent and honest in asserting Alice's control over A.

6.3 Trusting Issuers

Status: *Placeholder*

What defines issuer trust, i.e. what qualifies an actor to issue particular claims, is application specific (examples). Note that this is about the entity, not the pseudonym. We distinguish three methods for determining issuer trust:

- The issuer is a directly known trusted party such as a government, public organisation or acquaintance. This requires full identification of the issuing pseudonym.
- The issuer is not directly known/trusted but using external sources (such as a list of schools or register of doctors) which lists some public identity, we can trust it. This requires matching the pseudonym to the publicly used identifier. (full identification)
- The issuer is not directly known/trusted but its qualifications can be asserted in the system. This requires verifying the issuer, which we will discuss in Chapter 7.

6.4 Claim Meaning/Value

Status: *Placeholder*

We discuss the variability in meaning and value of claims, the difference between identification and qualification, and 1st hand vs 2nd hand validation.

Note that any two actors may agree upon meaning and value of claims, but this is much more complex in a multi-issuer and/or multi-verifier situation. This is why it should be at least supported in a CSL. Also, if we can improve the value of claims without permissioning (such as ESSIF is trying to do), we create a more affordable, open system.

6.4.1 Meaning

- In our scenario, a claim about the subject can describe any aspect of that subject that may serve as a qualification. For example, one may claim that Alice has a certain *age*, *skill*, *education* which may qualify her for a job, or one may claim that Alice *completed the marathon*, *got married*, *is entitled to health care*, *allowed to access a building.*
- The meaning of claims is virtually endless, so it cannot be defined in the semantic layer; i.e. the semantic layer should support any meaning of claims.

- It is however important that the issuers and verifiers of a claim have the same understanding of that claim. (Example?) For this problem, we created schemas.
- Schema creation is a common necessity so we should embed schema support in the semantic layer. The creation of schemas, hence definition of meaning, is left to the application layer.
- However, SSI argues that the Subject should be in control of her data. We have argued before that we cannot simply trust third party applications to mediate this control. Hence, the common independent Wallet should show Alice which data is requested and what she is about to share. Hence, she should have an understanding of that data. Hence it should be human readable. How?
- In this context, we usually refer to the claims as *attributes*, or aspects that apply directly to Alice. However, it may be important to provide information about other related subjects as well, because it may qualify Alice. For example: *owners or employees of restaurants get a discount at the whole sale.*[1].
- This is why Verifiable Credentials offers support for using linked data.

6.4.2 Value

- The value of a claim to Bob (the Verifier) is determined by how much it minimizes his risk in doing the intended transaction with Alice (not necessarily, he may just need to comply with KYC/AML rules).
- Accuracy. Let's first consider that Chris's claim may be inaccurate. In some cases this may harm Bob.
- When determining the value of a claim we see different degrees of abstraction. Claim-based identity is used because Bob cannot make the assertions about Alice himself, so he must trust other parties to do it for him. The probability and consequences of those claims being false may affect Bob. In practice, a multitude of methods is applied to minimize this risk to Bob, such as certification, standardization. At one end of the **spectrum**, Bob fully relies on Chris. At the other end, Bob wishes to gather maximum evidence from Chris that minimizes the likelihood that Chris's claim is false.
- Damages, Liability, Accountability. This is in turn determined by the damages that may be done by using the inaccurate information and subsequently who is responsible for that. Hence, with a claim can come obligations from an Issuer to a Verifier. In some cases this may fully exempt the relying party, Bob (for example, he may trust the accuracy of Dutch base registers, [but what about his damages?]). In other cases, Bob may

[1] bad example? because discount is actually meant for the company, not its reps

still be at risk. Hence, the value is influence by various factors. We will briefly dive into these.

- Authority. Apart from accuracy, in certain cases Chris must have authority to make the claim. Authority is a social (or legal) construct that defines who can do particular actions and who cannot. Assigning authority may be done to mitigate risk. However, this is usually an objective quality, within some legal/social framework. Hence, it is essential for Bob that if authority applies to the claim's issuer, that Bob asserts that the Chris is qualified. Authority may be held liable in some case.
- We first distinguish between the claim and the grounds, by saying that a claim is a statement that must/may be based on something, the grounds. These grounds may be an observation, such as an assertion. Bob may simply have a different idea of acceptable grounds (see problem e-IDAS of difference between countries defining LoA).
- Regardless of authority, the claim may still be inaccurate which may cause harm to Bob. We consider the following reasons for inaccuracy:
  - Issuer dishonesty. The issuer may lie about the claim.
  - Issuer incompetence. The issuer may be incompetent to determine the quality. Errors may be made. Data Quality.
  - Data Corruption.
  - Human/System Error.
  - Grounds. The method of assertion may not suffice (LoA).
- A second reason for inaccuracy is the fact that data may change. Claims are a derivative of the grounds, snapshots of aspects of the subject. However, many of these aspects may change over time.
- Also note that issuer qualification may change over time.
- Value to Bob is also defined by the extent of consent he receives from Alice. For which purposes can he use the data. Leave this out of scope!

6.4.3 Subject Binding

Discussed in different section? Contributes to the value.

6.5 Claim Naming/Formatting

Status: *Placeholder*

We discuss the need for and challenges with naming and formatting of claims. We use Zooko's triangle. We compare several approaches for using schemas.

6.6 Verifying

Status: *Placeholder*

We discuss the process of verification and how we can provide the parameters required by the cryptographic layer. We discuss verification is done primarily in service of some transaction or session, so we need to connect to that.

6.6.1 Claim Selection and Querying

Status: *Placeholder*

When a verification request is made, we need to find the claim(s) that match. If multiple matches, we must pick one. This requires a way of querying and selection. If unavailable, the claim must be collected at some issuer.

6.6.2 Disclosure Selection

Status: *Placeholder*

Optional: when disclosing a claim, we may limit disclosure to some proof or share the entire contents of the claim.

6.6.3 Claim Validation

Status: *Placeholder*

Given a successful verification, the verifier must still validate that the received information meets its requirements.

6.7 Issuing

Status: *Placeholder*

We discuss the process of issuing and how we can provide the parameters required by the cryptographic layer. We argue that issuing is primarily done in service of verification, hence must be done quickly without too much effort.

6.8 Common Actors

Status: *Placeholder*

TODO: *Move this section to the start of the chapter?*

6.8.1 Gate Keeper

Status: *Placeholder*

The gate keeper protects some resource or service from actors who are unqualified. Before granting access, the gate keeper requests verification of the subject according to some specification. Upon successful verification and validation, the user is granted access.

6.8.2 Data Store

Status: *Placeholder*

We model the data store as a set of mappings from some identifier to an attribute. This models for example many of the Dutch government's base registers (stelsel basisregisters), but fits many other cases such as customer registers, or attribute authorities in the access control domain. We assume that verifiers trust these data stores. Subjects should be able to retrieve their data in the easiest way

possible. The data store is usually protected by a gate keeper. We explain that this differs from free-form issuing which can be done in the portal paradigm.

Issue liability PK->BSN matching

6.8.3 Multi-Context Subject

Status: *Placeholder*

This models the common Alice, who is merely the subject in identity transactions across various contexts. He wishes to access services and claim his entitlements, which are all protected by gate keepers. He must therefore gather the necessary claims. These can come from various sources, but we will consider only the data store actor model as a source.

7 | Trust-by-Proxy

Status: *Placeholder*

This chapter addresses the problem that we cannot always trust Chris directly. For certain cases, such as the authorizations case described in Chapter 3.1, we have intermediate issuers that are not known or trusted directly, but eventually form a chain rooted at a trusted party. This chapter analyzes ways for the final verifier to verify the qualification of intermediate/proxy issuers.

7.1 Methods for Trusting Issuers

Status: *Placeholder*

We compare methods for trusting issuers:

- **Public Passive Verification:** The qualification of the proxy issuer is made public, so the verifier can check the proof without having to contact the proxy issuer.
- **Online Active Verification:** The verifier initiates a verification with the proxy issuer directly. The issuer must be online and consent to this verification.
- **Claim Forwarding:** The proxy issuer forwards the relevant qualification to the subjects it issued claims to. These subjects can then present these claims to the verifier.

7.2 Claim Forwarding

Status: *Placeholder*

This section describes different methods for claim forwarding based on the Stokkink and Pouwelse claim model, and Trustchain.

7.3 Proxy Issuer Responsibility

Status: *Placeholder*

The proxy issuer forms part of the chain of trust, which means his certainty of issuing contributes to the total certainty of the final subject of interest.

8 | User Control

Status: *Placeholder*

SSI's main promise is putting the user back into control. In this chapter we discuss how the user is in manual control over identity transactions. The user has several roles, but we focus on it being subject and also discuss signing things. The parties it interacts with for these purposes must identify themselves, making the user also a verifier.

8.1 User Flows

Status: *Placeholder*

We describe the different flows that the user is involved in, which makes clear when the user receives output and when it must provide input.

8.2 Understanding/Consent Sharing

Status: *Placeholder*

When a request for sharing comes in, it is the responsibility of the subject to judge this and consent or not. So?

8.3 Claim Resolution

Status: *Placeholder*

When a request for sharing comes in, the Wallet can lookup which claims satisfy the requirements. However, some claims may not be present. It is up to the user to provide these claims. We discuss alternative ways to resolve such request.

8.4 Manual Issuing

Status: *Placeholder*

In certain cases, the user must issue something. For example, when it authorizes another subject to do something. Also interesting for signing contracts/documents but this is out of scope. What at least remains is that, similar to sharing consent, the user should have full transparency and be well-informed before signing anything.

8.5 Manual Verifying

Status: *Placeholder*

In other cases, the user must verify something. Do we consider this in scope? Or can we safely defer this to external applications? Why?

8.6 Credential Management

Status: *Placeholder*

The user must manage his credentials. What does this mean?

9 | Implementation of Semantic Layer

TODO: *Write chapter Implementation*

9.1 IPv8

Status: *Placeholder*

This section explains our core dependency: IPv8 python implementation, how it fits the model of the cryptographic layer described in chapter 5. It also describes the REST API that is offered.

9.2 Tools

Status: *Placeholder*

This section describes the tools used: Typescript, React, Python for Android

9.3 IPv8 Agent Overlay

Status: *Placeholder*

We describe the light weight Typescript overlay we made for IPv8 and which functionalities it offers. It has the following components:

- **API Wrapper**: Wraps around the IPv8 REST API to make it easily accessible. Also offers a method for finding IPv8 peers to ensure they are reachable/online.
- **Attestor Service**: This service handles the role of Attestor. In our model, the Attestor and Attestee first reach consensus over the claim they wish to attest. The Attestor Service then allows to *stage* an approved attestation, such that when the Attestee sends its request over IPv8, the attestation can be immediately approved. It also offers a listening/event endpoint for incoming IPv8 requests that were not pre-approved.
- TODO: *Attestee Service*

  : Check whether the attestation was made correctly.
- **Verifiee Service**: Likewise, the Verifier and Verifiee reach consensus over what can be verified. The Verifiee may stage an approved verification, such that incoming requests may be immediately accepted. It also offers a listening/event endpoint for incoming IPv8 requests that were not pre-approved.
- **Verifier Service**: The Verifier Service is simply sugar over the IPv8 API that allows to trigger verifications and polls for their status.

9.4 Gate keeper

Status: *Placeholder*

9.4.1 Initiating

Status: *Placeholder*

9.4.2 Specifying Required Claims

Status: *Placeholder*

9.4.3 User Consent

Status: *Placeholder*

9.4.4 Performing Verification

Status: *Placeholder*

9.4.5 Performing Validation

Status: *Placeholder*

9.4.6 Hand-over to session

Status: *Placeholder*

9.5 Data Store

Status: *Placeholder*

This section describes our prototype of the data store/attestation server.

- We run a NodeJS server and IPv8 service in a Docker container.
- This attestation server can be configured in terms of *procedures*: predefined recipes that specify which claims are required to verify the Subject, followed by a specification of claims that may subsequently be issued to the Subject. For example, given a BSN, the server issues a KVKNR claim.
- For input of a procedure, the attestation server allows to connect to a local data source such as a database or API.
- This attestation server then allows the user and server to execute a protocol according to the spec of a procedure, using the IPv8 wrapper.

9.5.1 Service Discovery

Status: *Placeholder*

9.5.2 Configuring Procedures

Status: *Placeholder*

9.5.3 Connecting to Data Stores

Status: *Placeholder*

9.6 User Wallet

Status: *Placeholder*

9.6.1 Claim Management

Status: *Placeholder*

Users have to deal with a multitude of claims. We cluster claims into groups.

9.6.2 Wallet-Based Issuing

Status: *Placeholder*

Using the Data Store procedures as described above, the user can fetch data from known services. A screen is displayed for consenting to the verification, then the offered data is showed asking for a second consent. Finally the claims are issued.

9.6.3 Service Discovery

Status: *Placeholder*

How users discover new services, as they should not be hard coded in the wallet.

9.6.4 Trusted Parties

Status: *Placeholder*

How users can rely on/verify the identity of services.

10 | A First Application: Representing Legal Entities

TODO: *We now create an application that actually solves the original problem.*

10.1 Conceptual Design

Based on the problem as modelled in the previous section, we create a conceptual design of a solution.

- We have an app
- Everyone has same app
- 22

10.2 Mapping to Identity Primitives

10.2.1 Claim Information Model

- Full(P,L) issued by KVK, L:KVKNR
- PoA(P,L,J) issued by KVK, P:BSN (or Digid), L:KVKNR, J?
- Auth(P,L,J) issued by any person,

TODO: *We cannot make a claim type for every new J, so we must add that information in the meta somehow? The definitions of J are outside the system allowing for expansion.*

10.2.2 Verifying Strategies

Upon verifying Q(P,L,J), we must receive all evidence from the whole chain. We consider several alternatives:

Live Chain Check

Upon verifying P_n, the verifier verifies for all people $P_{i<n}$ that $Q(P_i, L, J)$. This raises the following issues:

- Verification depends on the whole chain to be online.
- Verification requires subject consent. This method would require the whole chain to spend attention on every verification. As this is impractical, we may solve this by automating the consent by the following rule: if any downstream subject grants a verification of $Q(P, L, J)$ to a verifier V, then automatically allow V to verify me with the same conditions.

This however makes it much easier to support revocation.

TODO: *Introduce $P_1, ...P_n$ notation above.*

Evidence Bundling

Alternatively, we forward all necessary evidence (all claims necessary to verify upstream peers) to the subject P_n. This eliminates the need to contact $P_{i<n}$.

10.2.3 Issuing Procedures

Full

The Chamber of Commerce identifies natural persons using their social security number (Burgerservicenummer, or BSN). Hence they need a way to map a subject P, identified by member id, to a known human.

TODO: *Continue*

PoA

TODO: *Continue*

Auth

TODO: *Continue*

10.2.4 Legal Entity Identities

TODO: *Two options: passive identity, active identity.*

▷ Person has key, or also legal entity, what about registers Which identities do we need to know/acknowledge?

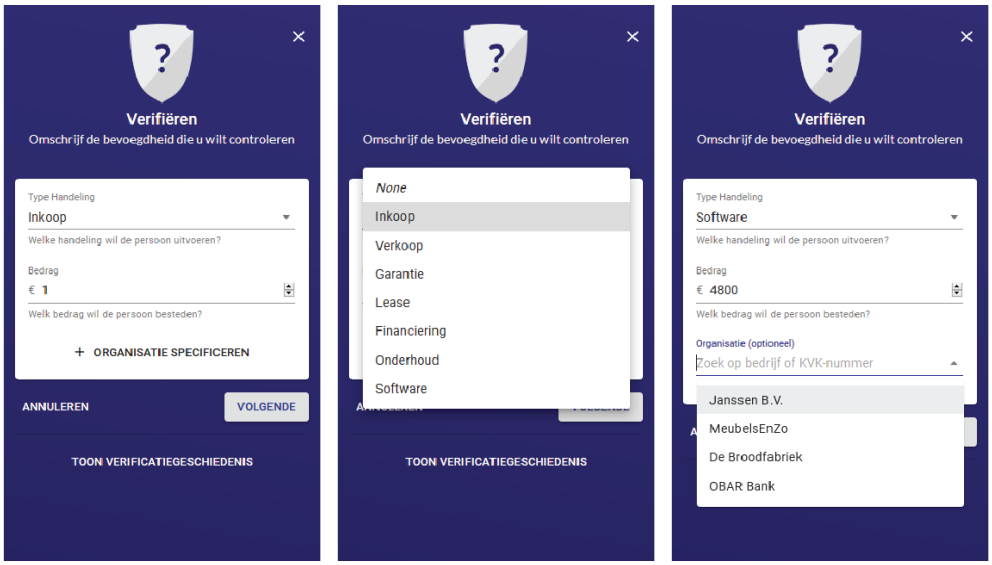
10.3 Design

TODO: *Write*

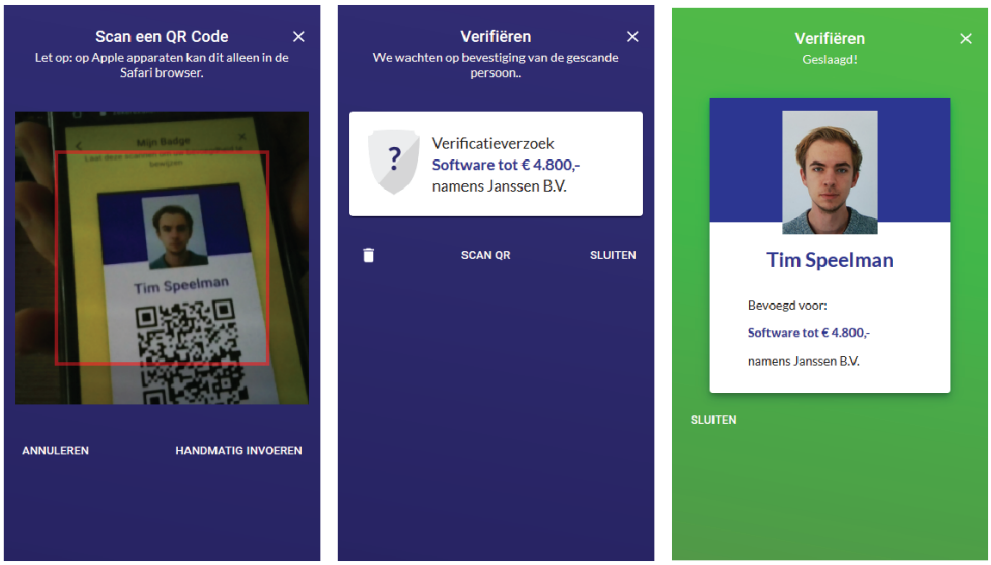
10.3.1 Home Screen

10.3.2 Verification Procedure

Figure 10.1 shows the Verification Procedure as seen by the Verifier. Figure 10.2 shows the same procedure, but from the perspective of the subject being verified.

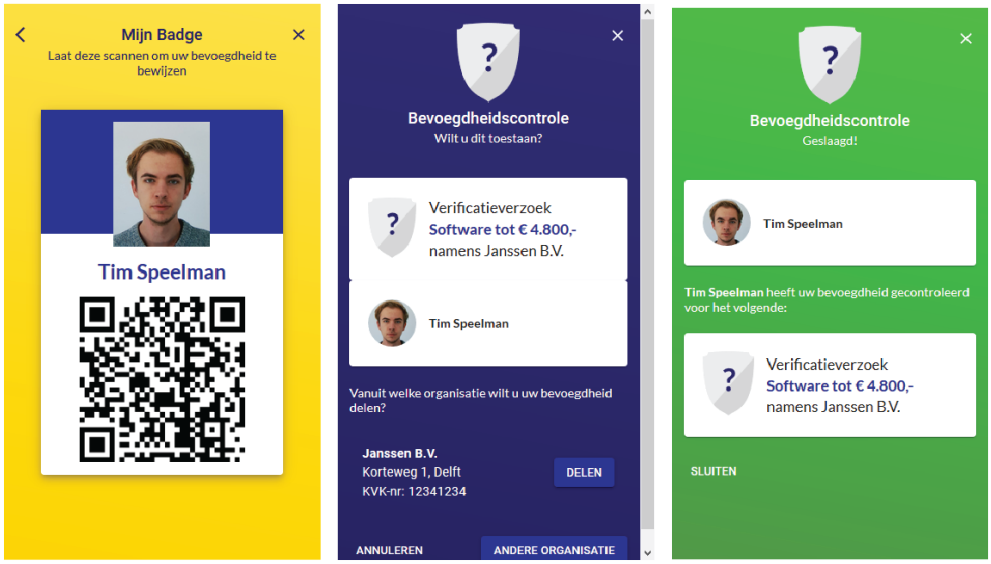


(a) New Verification (b) Select Activity Type (c) Select Organisation



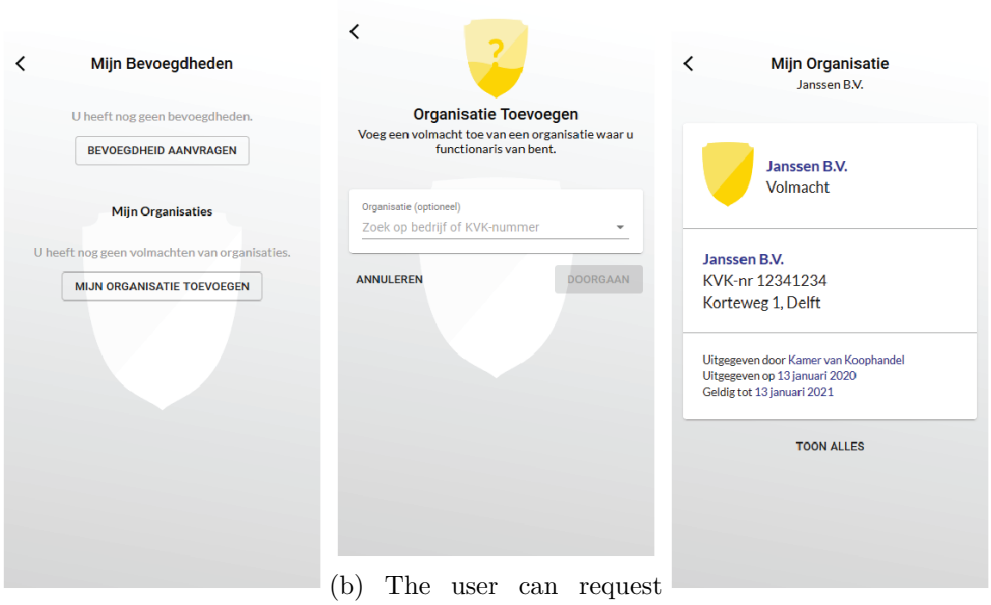
(d) Scanning a Badge (e) Await Response (f) Verification Succeeded

Figure 10.1: Zekere Zaken Verifier Flow



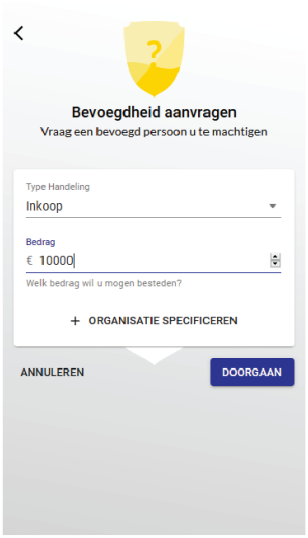
(a) Show Badge to Verifier (b) Received Request (c) Successfully Proven

Figure 10.2: Zekere Zaken Verifiee Flow



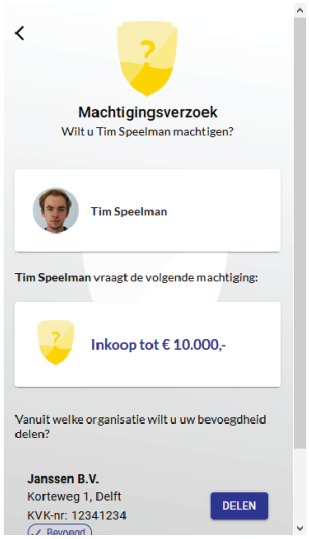
(a) List of Authorizations issued to the User (b) The user can request a Full-auth over an organisation. (c) Upon success, the issued Full-auth is stored.

Figure 10.3: Zekere Zaken Authorizations

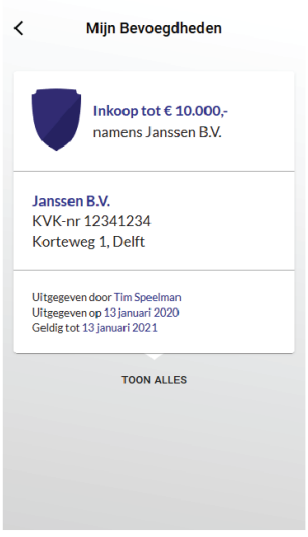


(a) The User can request a new Authorization

(b) The User can share this request via WhatsApp

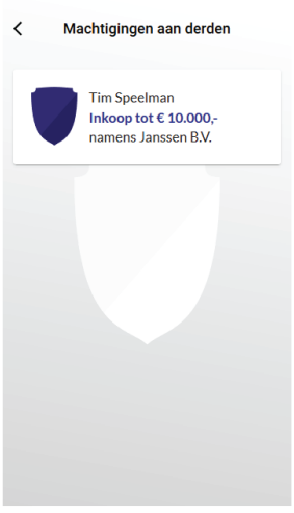


(c) The receiver sees the incoming request, along with the profile of the sender.



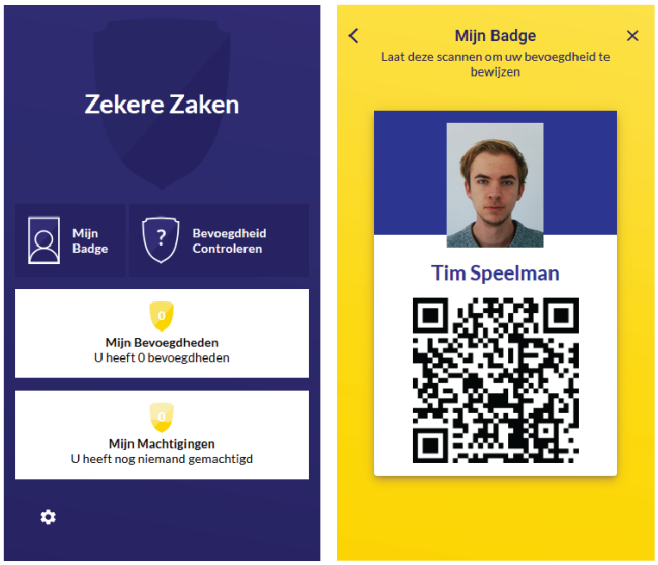
(d) After accepting, the Subject now sees a new authorization in his wallet.

(e) The Authorizer also sees the issued authorization.



(f) The Authorizer sees an overview of all issues authorizations.

Figure 10.4: Zekere Zaken Authorizations



(a) Home Screen (b) My Badge

Figure 10.5: Zekere Zaken Design

11 | Field Trials

TBD

12 | Conclusions

TBD

TODO: *Argue that this is more generically applicable*

13 | Future Work

TBD

Bibliography

[1] S. Azouvi, M. Al-Bassam, and S. Meiklejohn, "Who am i? secure identity registration on distributed ledgers", in *Data Privacy Management, Cryptocurrencies and Blockchain Technology*, J. Garcia-Alfaro, G. Navarro-Arribas, H. Hartenstein, and J. Herrera-Joancomartí, Eds., vol. 10436, Series Title: Lecture Notes in Computer Science, Cham: Springer International Publishing, 2017, pp. 373–389, ISBN: 978-3-319-67815-3 978-3-319-67816-0. DOI: 10.1007/978-3-319-67816-0_21. [Online]. Available: http://link.springer.com/10.1007/978-3-319-67816-0_21 (visited on 03/15/2020).

[2] Q. Stokkink and J. Pouwelse, "Deployment of a blockchain-based self-sovereign identity", *arXiv:1806.01926 [cs]*, Jun. 5, 2018. [Online]. Available: http://arxiv.org/abs/1806.01926 (visited on 10/08/2019).

[3] (). IRMA, Privacy by Design Foundation. Library Catalog: privacybydesign.foundation, [Online]. Available: /irma/ (visited on 03/30/2020).

[4] (). Sovrin.org, Sovrin. Library Catalog: sovrin.org, [Online]. Available: https://sovrin.org/ (visited on 03/30/2020).

[5] (). uPort.me, [Online]. Available: https://uport.me/ (visited on 03/30/2020).

[6] K. Cameron, "The laws of identity", p. 12,

[7] (). Verifiable credentials data model 1.0, [Online]. Available: https://www.w3.org/TR/vc-data-model/ (visited on 03/26/2020).

[8] (). IETF draft trustchain, [Online]. Available: https://tools.ietf.org/pdf/draft-pouwelse-trustchain-01.pdf (visited on 04/12/2020).

[9] (). IPv8 documentation — IPv8 1.9.0 documentation, [Online]. Available: https://py-ipv8.readthedocs.io/en/latest/ (visited on 04/12/2020).

[10] P. Otte, M. de Vos, and J. Pouwelse, "TrustChain: A sybil-resistant scalable blockchain", *Future Generation Computer Systems*, vol. 107, pp. 770–780, Jun. 1, 2020, ISSN: 0167-739X. DOI: 10.1016/j.future.2017.08.048. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X17318988 (visited on 04/12/2020).

[11] *Tribler/py-ipv8*, original-date: 2017-06-27T14:30:23Z, Apr. 11, 2020. [Online]. Available: https://github.com/Tribler/py-ipv8 (visited on 04/12/2020).

[12] (). IETF internet protocol, [Online]. Available: https://www.ietf.org/rfc/rfc0791.txt (visited on 04/18/2020).

[13] (). X.509 : information technology - open systems interconnection - the directory: Public-key and attribute certificate frameworks, [Online]. Available: https://www.itu.int/rec/T-REC-X.509-201910-I/en (visited on 04/18/2020).

[14] (). PGP user's guide, volume i: Essential topics, [Online]. Available: https://web.pa.msu.edu/reference/pgpdoc1.html (visited on 04/18/2020).

[15] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-DNF formulas on ciphertexts", in *Theory of Cryptography*, J. Kilian, Ed., red. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, and G. Weikum, vol. 3378, Series Title: Lecture Notes in Computer Science, Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 325–341, ISBN: 978-3-540-24573-5 978-3-540-30576-7. DOI: 10.1007/978-3-540-30576-7_18. [Online]. Available: http://link.springer.com/10.1007/978-3-540-30576-7_18 (visited on 04/18/2020).

[16] K. Peng and F. Bao, "An efficient range proof scheme", in *2010 IEEE Second International Conference on Social Computing*, Aug. 2010, pp. 826–833. DOI: 10.1109/SocialCom.2010.125.

[17] D. Benarroch, M. Campanelli, D. Fiore, and D. Kolonelos, "Zero-knowledge proofs for set membership: Efficient, succinct, modular", p. 47,

[18] *eIDAS regulation (EU) no 910/2014 of the european parliament and of the council of 23 july 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing directive 1999/93/EC*, Legislative Body: EP, CONSIL Library Catalog: EUR-Lex, Aug. 28, 2014. [Online]. Available: http://data.europa.eu/eli/reg/2014/910/oj/eng (visited on 04/16/2020).

[19] P. A. Grassi, M. E. Garcia, and J. L. Fenton, "NIST digital identity guidelines: Revision 3", National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-63-3, Jun. 22, 2017, NIST SP 800–63–3. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-3.pdf (visited on 04/04/2020).

[20] C. Allen. (). The path to self-sovereign identity, [Online]. Available: http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html (visited on 10/10/2019).

List of Todos

Section Status