[DRAFT] Industry-Grade Self-Sovereign Identity

On the Realisation of a Fully Distributed Self-Sovereign Identity Framework

Rowdy Chotkan and Johan Pouwelse
r.m.chotkan@student.tudelft.nl
j.a.pouwelse@tudelft.nl

The internet was created without a standardised identity layer, resulting in each user having to manage a plethora of digital identities which hold no legal value, often requiring cumbersome identity card checks, e.g., through digital photocopies. Initiatives such as *User-centring identities* have mostly failed, resulting in asymmetrical control over our digital data held by Big Tech. Self-Sovereign Identity (SSI) can prove to overcome these hurdles. SSI aims to put one at the centre of their digital presence, making them the owner of their identity. This enables full control over your own data and opens up the possibility of legally valid digital identities. We present Industry-Grade Self-Sovereign Identity (IG-SSI): a fully distributed SSI framework, requiring no specialised nodes or hardware, in which equality and offline usability are at the core of the design. The resulting schema allows for attestation signatures, presentation, verification, and revocation through Zero-Knowledge Proofs (ZKPs). Fully distributed revocation is achieved through the Hybrid Revocation Model (HRM): a gossip-based revocation model enabling offline verification. The HRM shows improvements with respect to already presented revocation designs and portrays great scalability. IG-SSI has been validated through field labs and has been designed in collaboration with the Dutch Ministry of the Interior and Kingdom Relations, we hope that SSI in general gains traction and, as such, a legally valid Self-Sovereign Identity may soon be deployed.

Introduction

SINCE the dawn of the Information age, digital trust has been an issue requiring many workarounds. The core concepts of the internet are simply not built with trust in mind: there exists no standardised identity layer. As a result, the current landscape of identification and authentication mechanisms form a digital ecosystem of "digital one-offs" (Cameron, 2005). As a consequence, the popularity of identity management solutions by pioneers of the Internet has resulted in an oligopoly in digital identity of Big Tech companies. Wherein a regular oligopoly consumers are at a disadvantage price-wise (Stigler, 1964), in this technical oligopoly the identity providers have an asymmetrical control of ones digital presence and have gained the ability to nullify access to numerous services in case one violates their terms of service. In addition, this oligopoly results in large information asymmetries, as Big Tech has increasing amounts of knowledge on their users.

Furthermore, increasing needs for digital identities from governments such as the European Union, has catapulted the research into and relevancy of the field itself. With the State of the Union 2020 address by President Von der Leyen portraying the relevancy of the problem:

"*Every time an App or website asks us to create a new digital identity or to easily log on via a big platform, we have no idea what happens to our data in reality. That is why the Commission will soon propose a secure European e-identity. One that we trust and that any citizen can use anywhere in Europe to do anything from paying your taxes to renting a bicycle. A technology where we can control ourselves what data and how data is used.*"

The need for digital identity, furthermore, stems from the urgency of COVID-19 vaccination passports, requiring digital verifiability and validity across borders. As a result, this digital and socio-economical gap can prove to be filled by the concept of *Self-Sovereign Identity* (SSI). SSI aims to generate digital trust by providing verifiable digital identities, putting the user at the centre. SSI is a concept requiring multiple state-of-the-art technologies to be realised, hence, the feasibility of developing a schema that is both technologically and usability-wise sound, can be proven to be hard. Several solutions exist (e.g. Sovrin[1] and Serto[2], however, many require proprietary technologies or hardware, specialised infrastructure limiting equality in the network, or do not provide academic substantiation for their claims. As SSI combines multiple technologies, such as decentralised ledger infrastructure, public key infrastructure, and secure

[1] For Sovrin, see: `https://sovrin.org/`

[2] For Serto, see: `https://www.serto.id/`

data management, many of the existing solutions do not stem strictly from academia, making their results more difficult to reproduce and limiting the analysis of their design choices.

This article introduces *Industry-Grade Self-Sovereign Identity*: a purely academic Self-Sovereign Identity framework focusing on an open standard, with intrinsic equality across the network, an offline-first design, and capabilities of fully distributed revocation. The scheme is based on the previous works by Stokkink & Pouwelse (2018), Stokkink et al. (2020) and builds upon the IPv8 protocol stack (Halkes & Pouwelse, 2011; Zeilemaker et al., 2013). The main contributions of this work are a functioning SSI scheme, which can be said to be of *industry-grade*. IG-SSI makes the following contributions to the work set out by Stokkink & Pouwelse (2018): (1) Trusted Authority (TA) concepts, (2) offline verification capabilities through Offline Revocation Lists (ORLs) , (3) revocation mechanisms (referred to as the Hybrid Revocation Modal), (4) improved security and usability considerations (6) a reference implementation of the semantic layer, and (7) reference implementations showcasing practical use-cases. Furthermore the solution has been validated using a field test.

The article is structured as follows: firstly we discuss the design of the framework, introducing relevant concepts and design considerations. Secondly we discuss the resulting schema, its implications, and the evaluate the results. Finally, we discuss related works.

Design

Self-Sovereign Identity is build around the notion of *Attestations*. As such, we require three main behaviours: firstly, attestations signing, secondly, attestation presentation and verification, and finally, attestation revocation. In this section, we firstly discuss the Attestations themselves, then the signing and presentation flows, and finally, revocation.

Attestations

Attestations can be said to be the core concept of Self-Sovereign Identity. With attestations, we refer to cryptographically signed data, enabling verification of information through validation of signatures. In other words, a client, i.e. an Authority, cryptographically signs—attests to—information for another party, the Subject. Allowing any third-party, a Verifier, to verify that the data was attested to by the Authority. As becomes apparent from this description, these roles are neither mutually exclusive nor static: a single party can both be e.g. the Authority and the Subject for an attestation, whilst being solely a Verifier in another instance.

Asymmetric Encryption

A rather straightforward realisation of Attestations can be achieved through asymmetric encryption and signatures. For instance, using public key encryption, an Authority can, through the use of his private key (SK), encrypt the hash of a plaintext message (m) and the public key of the Subject (PK), resulting in $e(\mathcal{H}(m|pk))$. This allows any party that knows the corresponding public key of the Authority, to verify that the data m was attested to by the Authority for the Subject. There, however, exist several limitations with this approach. Firstly, disclosing the attestation and, thus, verifying the signature always reveals the corresponding plaintext values. This is not desirable, as the attestation may comprise sensitive data. Secondly, this would disclose more information than is necessary. For instance, verifying whether one is of age of majority, should not require the disclosure one's actual age. Rather, proving that one is above said threshold should suffice in such an instance. As such, Zero-Knowledge Proofs (ZKP) may prove to overcome these hurdles.

Zero-Knowledge Proofs

ZKPs allow the verification of a value without disclosing the value to the Verifier Smart (2016). ZKPs especially enable the integration of the minimisation property of IG-SSI. Broadly speaking, there exist two types of Zero-Knowledge Proofs: (1) exact proofs and (2) range proofs, both of which can have interactive or non-interactive variants. We propose the usage of ZKPs for their added benefits of non-disclosure and range proofs. For regular static values exact proofs should be used, whilst any form of attestation requiring a number, range proof should be used.

Attestation Design

We propose a design based on the work set out by Stokkink et al. (2020). The attestation procedure is visible in Figure 5. The design uses multiple phases, with optional steps. We make the distinction between two types of attestations **[TODO:** Rename "value-attestation request" to proof-request?**]**:

1. *Value-Attestation*: this type of attestation can be said to be the core type. It is responsible for incorporating a specific value into a Zero-Knowledge Proof. The verifiable-nature of attestations stems from this type. As visible in Figure 5, the design of this attestation allows for multiple *proof formats*, allowing for flexible selection of ZKPs and, thus, attestations. This disallows the lock-in of specific proof types, as any client can propose the usage of any type of proof, which can be used as long as the corresponding Authority supports the proposed type as well.

2. *Credential-Attestation*: this type of attestation is a reference to a Value-Attestation. This secondary type of

> attestation allows for the subsequent attesting of values, through attestation chaining: subsequent authorities can attest for the same value by attesting to the Value-Attestation as opposed to requiring a separate Value-Attestation. Credential-Attestations also refer to metadata, which allow for validity terms and sign dates.

There exist several benefits to this construction. Firstly, the aforementioned chaining of attestations allows for multiple authorities to attest to a value. As such, real-life signature scenarios can be modelled through attestations. This allows for concepts such as *segregation of duties* and other shared responsibility scenarios, in which multiple parties must attest for a certain claim in order to be valid. For instance, a credential attesting for the ownership of a driving license, may require a signature by both a government body handing motor vehicles and a local government. The ability for multiple attestations for a single value can prove to be capable of handling such real-life scenarios. Secondly, subsequent attestations do not require the knowledge of the plaintext value. For instance, continuing on the driving license example, a local government does not require extensive knowledge on the license itself, a signature by the responsible government body should be enough for them to attest. As a consequence, this aids in data minimisation on subsequent Authorities. Finally, in case of attestation properties such as validity terms, a renewal of an attestation can simply be a new Credential-Attestation for the Value-Attestation, not requiring the re-attestation for the actual data. Again, this aids in data minimisation and privacy, as the plain text values do not have to be disclosed. Additionally, different Authorities can adhere to different metadata of the same attestation without influencing other parties. By allowing different Credential-Attestations for the same Value-Attestation, different metadata is enabled to exist for the same Value-Attestation. For instance, different Authorities can set different expiration dates on the same Value-Attestation. Again, when the expiration date has passed, the issuing Authority can simply re-attest for the same Value-Attestation, generating a new signature for the Credential-Attestation.

Attestation Flow

The attestation flow consists of two phases, the Proof-phase and the Credential-phase which do not always require subsequent execution. More specifically, for a single to be attested claim, the Proof-phase requires a single execution, which must occur before the Attestation-phase. Whilst subsequently, the Credential-phase can be performed indefinitely.

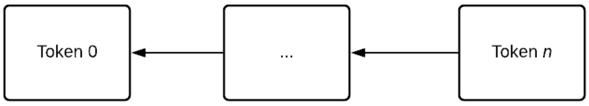

Figure 1

Token Chain

Proof-phase

The Proof-phase is initiated by a Subject. A Subject aims to have a claim attested to by an Authority. It does so by requesting an Attestation from the Authority. In this request, the Subject must make the attribute name, the to be used proof format, and his public key apparent. This public key, is a one time used public key, of which the private key must be stored by the Subject. The usage of single-use public/private key pairs, allows for additional privacy properties imposed on the system, which will be explained in **[TODO:** Add reference + write small section on this subject**]**. Additionally, any other information that is to be known by the Authority must be sent along, for instance the requested plaintext value. Note that the value is, thus, not required to be sent by the Subject. The implication of this, is that an Attestation can be made for the Subject, without the Subject knowing the exact value. This, hence, allows for the secure storage of information, in the form of a ZKP attestation on a client, without the actual revealment of the underlying value.

The receiving Authority may respond to the request, making him an issuing Authority. The Authority generates a Value-Attestation of the type defined by the *proof format*. This attestation, thus, incorporates the value belonging to the requesting attribute name. This attestation is sent back to the requesting Subject. After having received the Value-Attestation, the requesting Subject moves onto the *Credential-phase*.

Credential-phase

In the Credential-phase, a requesting Subject requests an attestation for a certain Value-Attestation, making it a Credential. It does so by disclosing all already attested Credential-Attestations belonging to the Credential. The core of each Credential is an Attestation Token. Each Token contains the hash of a Value-Attestation and points to the previous Token. This has been visualised in Figure 1. The first token, comparable to a genesis-block in Blockchain structures such as that by Nakamoto (2009), contains the hash of the public key belonging to the Subject. Any subsequent Credential, thus, generates a new Token, occupying a place as a shackle in the chain. When an Authority is requested to attest to a Credential, it may request each previous token and, thus, the hashes of each previous attestation, after which it can verify these attestations. As such, it is improbable for a client to attempt to hide the existence of an attestation or attempt to cheat the system, as otherwise the attestations of other Authorities become invalid (as the hash of the token will no longer be correct). Hence, as visible in the second phase of in Figure 5, after having received a Credential request, the Authority may request any missing tokens until he gains confidence to attest for the Credential, creating a Credential-Attestation. Note that these Tokens do not reveal any information about the underlying Value-Attestations, as they merely contain the hash value. When an Authority attests to a Credential, it generates a signature for the hash of the corresponding metadata, which in turn points to a Token. This structure of referencing data structures is visualised in Figure 2. As visible, a Token refers to a single Value-Attestation. However, multiple metadata instances may reference a single Token and, similarly, multiple Credential-Attestation may reference a single metadata instance. These relationships allow for the aforementioned properties and scenarios. As becomes apparent from this description, the second phase, i.e., the Credential-Phase, can thus be repeated indefinitely as numerous Authorities can co-attest and re-attest for an Attestation.

Verification Flow

In order to verify attestation values, a presentation procedure must exist. As clients may decide themselves whether to share attributes, we propose the structure as visible in Figure 3 **[TODO:** Add token requests**]**. In this structure, an Authority requests an attribute with a specific name. A Subject may subsequently decide whether to respond to such a request and to disclosure the corresponding attribute. Note here that the credential request is not necessarily required, as a client can disclosure an attribute directly. However, the specification of an attribute name, aids in selective disclosure, whilst additionally allowing the Authority to determine whether a specific credential is solicited. After a credential has been disclosed and, thus, presented, the Authority may verify its validity.

Verification

We propose two types of verification. Firstly, an interactive variant and, secondly, a non-interactive variant, enabling offline verification. The general flow of the interactive variant is visible in Figure 4. For active verification, an Authority requests the underlying Value-Attestation by presenting the attestation hash to the Subject. The Subject may consent through sending the requested Attestation. After the Authority receives the ZKP commitment, the Authority may send challenges to verify the underlying value. Note that for this to happen, the Authority must either be already aware of the value belonging to the attribute or the plaintext value must be shared. Sharing of the plaintext value can be done during presentation-time. This should be performed using encryption in order to preserve privacy, for instance through the use of RSA by Rivest et al. (1978). The second method for verification uses the attestations made by other authorities. In order for this attestation to pass, the list of attestors must contain an authority that is trusted by the Verifier. If this is the case, a Verifier may accept the value proposed by the Subject in case the metadata contains the hash of this value and the signature made by one of the acknowledged authorities over the metadata is valid. This approach does not require any connectivity between the Subject and Verifier, apart from the presentation itself. However, a presentation does not necessarily require any form of digital communication (e.g. through QR-codes). It is, however, to note that this offline verification, thus, does not rely on any additional token requests and, as such, all tokens must either be made directly apparent to the Verifier during presentation-time or the verifier must make its decision based on the presented Attestation and his reliance on and knowledge of acknowledged authorities.

Revocation

[TODO: too deep, some information should be omitted**]** Revocation is one of the main unsolved issues in Self-Sovereign Identity and an issue in distributed systems as a whole. As in real life contracts and other agreements may become invalid before their termination date, the ability to revoke attestation in SSI must be available as well. Several motivations exist for revocation:

- Erroneously signed data: in case data was signed accidentally.
- A Legally invalid contract: in case at a later instance it became apparent that the signed data can not be legally upheld.
- Premature termination of a contract: in case a certain breach of contract occurs.

Note that expiration is not one of these listed motivations, as time-bound attestations can be realised using signed metadata. It is important that revocation can never occur due to expiration, as some claims should never be able to be revoked. For instance, it should not be possible for an authority to revoke a signature indicating someone is of legal age (unless in the rare instance that it was erroneously signed and can be publicly verified that this was, indeed, the case), as this fact can never become false.

As IG-SSI is built without specialised validation nodes, present in some blockchain-based protocol such as Zhou et al. (2019), there is no trivial non-interactive solution of revocation of signatures. The trivial solution is to actively query

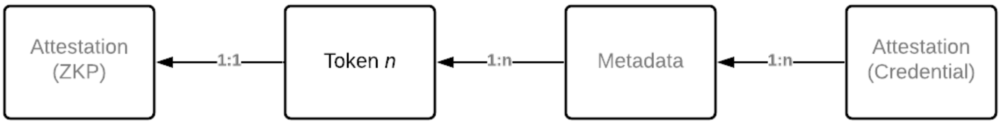

Figure 2

Data Structure Relationships

signees (i.e., the responsible authorities) and verify that they still attest for the signed information. There exist multiple problems with this solution. Firstly, this querying requires interactivity with the signee(s) of an attestation. Whilst interactivity is not a problem per se, it does introduce additional overhead. It requires the signee(s) to be online. Whilst availability often is a key characteristic in distributed systems, there is no guarantee that specific clients, i.e. the Authorities, are available. Additionally, this interactivity generates overhead in the verification process: apart from challenging the presenting client, the signees have to be actively queried, introducing additional verification time and network traffic.

Secondly, as a requirement for enabling this interactivity, a (network) connection to the signees must be available. This completely nullifies the possibility for offline verification. Next, we discuss our solution for revocation: *The Hybrid Revocation Model* (HRM). This model requires no additional interactivity during verification and enables offline-verification.

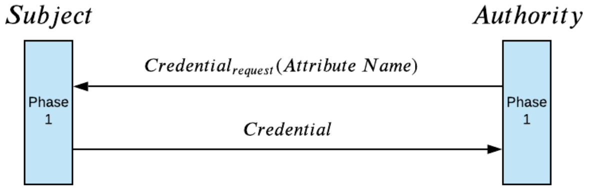

Figure 3

Attestation Presentation

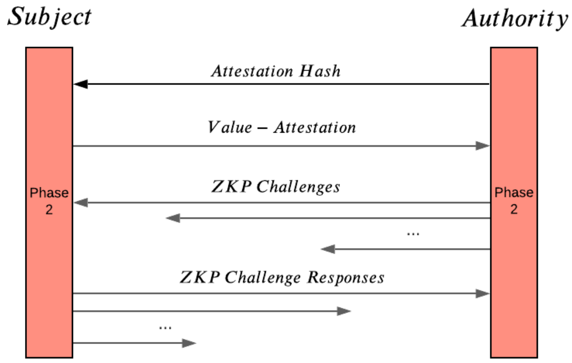

Figure 4

Interactive Verification

Hybrid Revocation Model

The Hybrid Revocation Model attempts to overcome the hurdle of interactivity whilst allowing for flexibility, enabling offline-verification. IG-SSI is fully distributed and as such, each node is equal. As a consequence, the client performing the verification must be aware of any revocations belonging to a presented attestation. Selecting specific nodes for distributing and holding revocation, would deteriorate the equality principle. As these nodes would, then, possess the ability to hide certain revocations from the network or could lead to collusion (Khovratovich & Law, 2017). As such, revocations should be public data. I.e., every revocation should be visible to every client. The *hybrid* nature of the model, stems for its offline capabilities: during verification-time, clients do not require to be online. They merely require occasional synchronisation of revoked attestations through communication with other peers.

In HRM, each peer has the possibility to posses the same information about revocations. Revocations are propagated through the network, enabling each peer to store revocations from clients they trust. This concept builds upon the notion of Trusted Authorities. The general flow of the design can be seen in Figure 6. The protocol has three key concepts:

1. Trusted Authorities (TAs)
2. Propagation
3. Offline Revocation List (ORL)

Next, we explain each concept.

Trusted Authorities

In a fully distributed setting, client are responsible for their own actions. Meaning that revocation are as meaningful as the extend to which they are used by the clients. This property makes it that clients themselves are able to acknowledge

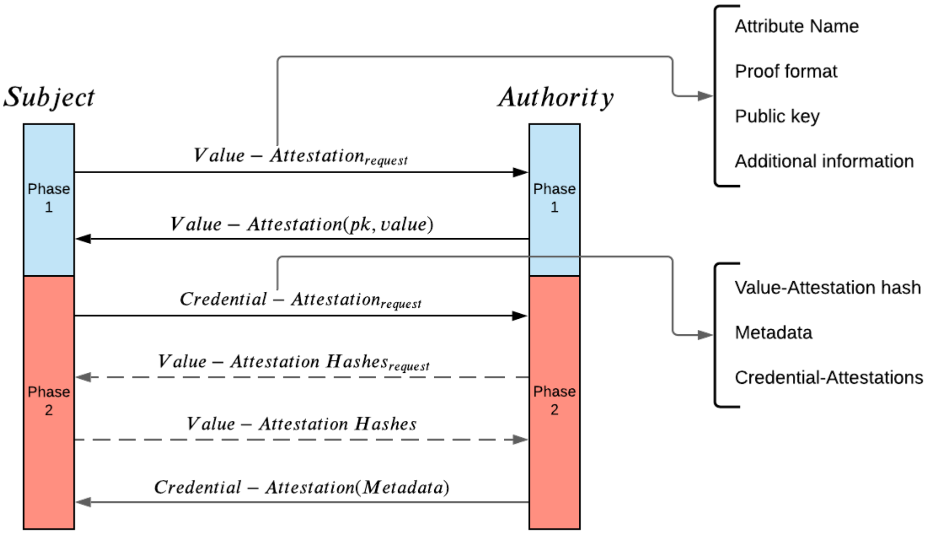
Subject
Authority
Phase 1
Phase 2
$Value - Attestation_{request}$
$Value - Attestation(pk, value)$
$Credential - Attestation_{request}$
$Value - Attestation\ Hashes_{request}$
$Value - Attestation\ Hashes$
$Credential - Attestation(Metadata)$
Attribute Name
Proof format
Public key
Additional information
Value-Attestation hash
Metadata
Credential-Attestations

Figure 5

Attestation Flow

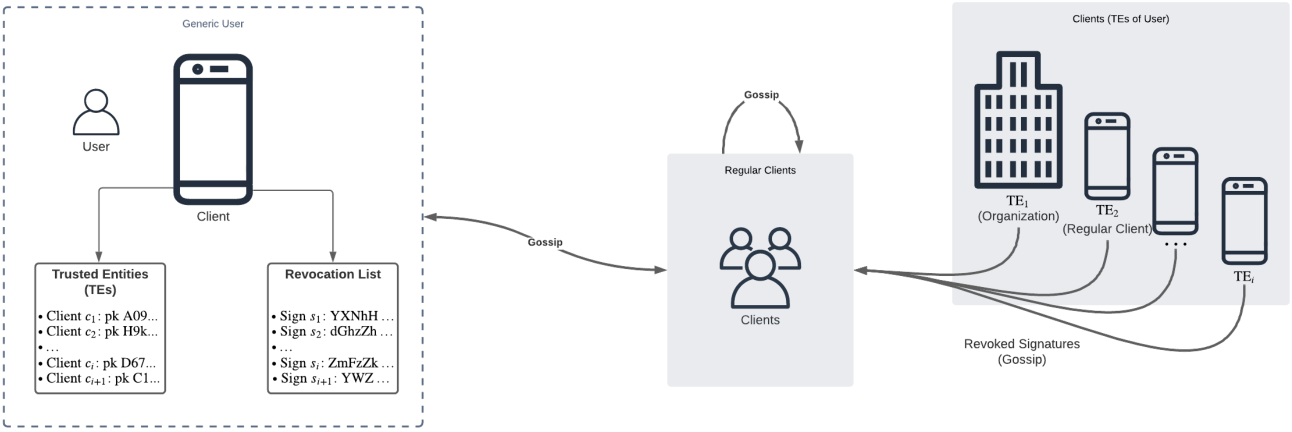

Figure 6

The Hybrid Revocation Model (HRM)

or reject revocations. A criterion on which a client is able to determine the validity of a revocation is whether the Revoking Authority is trusted by the client. This is where we introduce the notion of Trusted Authorities (TAs). As mirrored by real life, a person has (relatively speaking) a choice whether to acknowledge a certain authority. With SSI aiming to be a digital extension to one's identity, one should also be able to make such an acknowledgement in the digital domain. As an added benefit, identification in the digital domain can prove to be more verifiable than physical verification. We propose the usage of a Trusted Authority Storage (TAS). In the TAS, the public key and the public key hash of a TA are stored. We make the distinction between acknowledged (trusted) and Unacknowledged Authorities (UAs). As discussed previously, client roles are neither static nor mutually exclusive. As a consequence, potentially every client can be an Authority. However, it is up to a client to determine whether an authority is a TA or an UA. In terms of distributed revocation: a client aims to accept only those revocations of which he knows that he can trust the authenticity. The results of acceptance are the storage of the revoked signatures and propagation towards network.

Propagation

In order to safeguard availability in the network and enable offline verification, we propose the propagation of revocations throughout the network. This requires two means: firstly, a verifiable revocation format and, secondly, a propagation protocol for the revocations. We propose the structure as visible in Table 1. This design, in addition to the revoked hashes, includes a public key hash, a version number, a specification for the used hashing algorithm and a signature. The public key hash allows for the retrieval of the public key in case said key belongs to a TA acknowledged by the receiving client. This public key can, thus, be retrieved by querying the TAS. In case the public key belongs to a TA, the signature can be verified by concatenating the version number with the revocations. Unique version numbers allow clients to ensure that they are either fully synced with the network or are missing certain revocation versions. The revocations themselves are to be the hashes belonging to the attestation metadata. This, thus, invalidates any attestations made to this metadata and the token it points to. As a benefit, this reduces overhead when presenting attestations as solely based on the metadata, an attestation can be deemed to be valid or revoked. The hashing algorithm specification improves the transparency and robustness of the schema. For instance, hashing algorithm recommendation may differ in the future due to e.g. efficient collision finding. Allowing for specification enables the interchanging of this algorithm, aiding future-proofness and flexibility.

The propagation itself requires a protocol that ensures information is (eventually) spread across the entire network, whilst also ensuring that unavailable nodes receive the information at a later instance. For this, we propose the usage of gossip protocols with interval re-transmission. Gossip protocols are communication protocols which allow for the periodic exchange of data with (random) peers (**?**). The periodic exchange of data with peers, makes gossip protocols a prime candidate for the realisation of distributed revocation. Furthermore, in order to decrease the overhead of gossiping a theoretically unbound number of signatures, we propose the usage of a multi-step update procedure. This procedure has been visualised in Figure 7. This procedure is split-up in two phases: firstly, a gossiping client gives notice to a client that it possesses specific authority-version pairs, containing the public key hash of an authority and the latest version it is aware of. Next, the receiving client can request an update by sending back the latest versions of the revocations stored in their TAS. This allows a client to selectively send updates, as the receiving party makes an underbound of the known versions apparent. This extra step of selective requesting relieves a large amount of data as clients are not necessarily interested in revocations by certain authorities as they may be considered UAs or a client may already be fully synced.

We note that this procedure may be fine-tuned through the usage of revocation dates. Revocation dates may allow clients to opt out of old revocation versions, optimising storage usage as old revocations may no longer be relevant in the system due to the validity terms of the attestations having passed.

Offline Revocation List

Any valid received revocation should be stored by a client for later reference. The storage of revocations allow for offline (in)validation of attestations. This storage we deem the Offline Revocation List (ORL). Whilst no specific storage structure is required, we do propose the usage of Bloom filters for member checking. A Bloom filter is a memory- and time-efficient probabilistic data structure, which allow for efficient membership operations (Bloom, 1970). Raya et al. (2007) discuss the benefits of Bloom filters in Certificate Revocation Lists (CRLs), which can be transformed to our concept of ORL, as the ORL can be deemed a more generic variant of a CRL.

As yearly up to 340.000 identity documents are stolen in a country as The Netherlands, the same amount of revocations must be possible on a year basis (Nieuwsuur, 2019). As such, revocation membership checking can prove to become quite expensive both memory- and runtime-wise. Even with the most efficient algorithms such as Binary search, with a runtime complexity of $O(\log(n)$, the execution time of such a search can be too long, usability-wise. As such, we propose the usage of membership verification through Bloom filters, in which a membership search on the actual data is only performed in case of a possible match. Additionally, it can be said that the probability of encountering a revoked attestation should be extremely unlikely. As we assume the majority of the nodes to be honest, they have no incentive to attempt to cheat the system. As such, Bloom filters with their property of ensuring an item has no membership in case the filter does not contain it and, thus, only having to validate using the actual data in case the filter may contain the item, Bloom filter can prove to achieve much stricter execution timings for validation.

Furthermore, we note that the ORL can be replaced by a Bloom filter entirely. A client may chose to accept the probabilistic nature of Bloom filters over the exact membership check from memory. Such nodes may not be able to aid in the propagation of the revocations, however, the low memory requirements may prove to make the protocol suitable for IoT devices.

Analysis

For the realisation of IG-SSI, we implemented three semantic layers, namely:

1. **Attestation Layer**: abstracts the signing of Zero-Knowledge Proofs and verification.

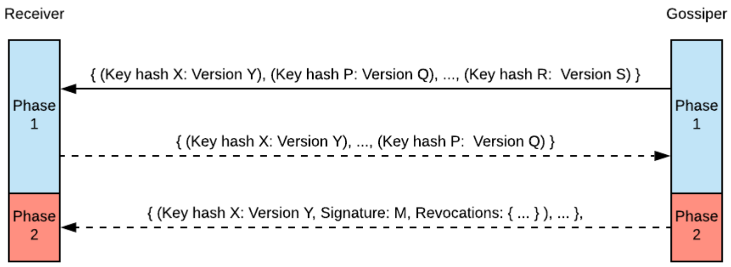
Receiver
Gossiper
Phase 1
Phase 2
Phase 1
Phase 2
{ (Key hash X: Version Y), (Key hash P: Version Q), ..., (Key hash R: Version S) }
{ (Key hash X: Version Y), ..., (Key hash P: Version Q) }
{ (Key hash X: Version Y, Signature: M, Revocations: { ... }), ... },

Figure 7

Multi-step Update Procedure

Table 1

Verifiable Revocation Update Format

Authority key hash	5e2bf57d3f40c4b6df6...
Version	1701
Hashing Algorithm	SHA3-256
Signature	422c06fbb4fbd23d33...
Revocations	b788c5b28dba2fc6a0... 7f2519609cf157d7e9... ... e2d7610dcb53724675...

2. **Credential Layer**: abstracts the attestations of Authorities over ZKPs and enables chaining of attestations.

3. **Revocation Layer**: abstracts the handling of revocations over credentials.

These three layers are built on top of the academic communication protocol of IPv8[3] primarily based on the works by Zeilemaker et al. (2013) and Halkes & Pouwelse (2011). The selection of IPv8 stems from firstly its academic background, proving its viability through various publications. Secondly, IPv8 allows for direct client-to-client communication, hence, enabling a fully distributed infrastructure at the core of the solution. Finally, IPv8 does not require (expensive) Proof-of-Work algorithms utilised by Blockchain structures such as Nakamoto (2009) and Buterin (2013). In addition to the aforementioned semantic layer, a secure multi-party communication channel has been developed.
The code for the reference implementation of these semantic layers is available on the IPv8 repository[4]. Additionally, a mobile client, in the form of an Android application, has been deleveloped[5].

The Semantic Layers

Next, we discuss additional implementation details for some of the semantic layers.

Attestation Layer

The *attestation*-layer abstracts the logic for signing and verifying the ZKPs used. Whilst not necessarily requiring ZKPs, as the design allows specification and, thus, negotiation of used proof formats, ZKPs are highly recommended due to the intrinsic properties they introduce to the system. The design itself is, thus, proof-agnostic as one can implement any type of proof. Per choice, two types of ZKPs are implemented. Firstly, a ZKP proof allowing arbitrary data and the verification of exact values. For this, the algorithm proposed by Boneh et al. (2005), allowing verifiable computation through 2-DNF formulae over bits. Boneh et al. (2005) is a homomorphic public-key encryption scheme which allows for universally verifiable computation, a property which is desirable in Self-Sovereign Identity. Additionally, this allows for interoperability with the schema proposed by Stokkink et al. (2020). Secondly, the range ZKP proposed by Peng & Bao (2010) has been implemented. This ZKP allows for the encoding of integer values laying in a specific range. Peng & Bao (2010) requires constant costs, proving to be more efficient than previously proposed solutions. We implemented the commitment scheme proposed by Boudot (2000) in order to realise the range proof by Peng & Bao (2010). Both of these proofs are interactive. However, as shown by Koens et al. (2018), the schema introduced by Peng & Bao (2010) can be made non-interactive.

[3] For the official (Python) documentation of IPv8, see `https://py-ipv8.readthedocs.io/en/latest/`

[4] For the Kotlin IPv8 repository, see: `https://github.com/Tribler/kotlin-ipv8`

[5] Fr the Android application, see: `https://github.com/Tribler/trustchain-superapp`

Credential Layer

The *credential*-layer has been implemented as per design specification.

Revocation

For revocation, we implemented a custom gossip protocol. For the ORL, a Bloom Filter (Bloom, 1970) has been implemented for memory-usage and run-time improvements. Based on the expected 300.000 lost identification documents per year, as presented by Nieuwsuur (2019), the following memory and time considerations can be made. Firstly, a storage for 300.000 hashes of 32 bytes each, results in a space usage of at least 9.2 megabytes. Whilst a Bloom filter with a probability of a false positive of 1 in 100 million and 27 hashing functions, can achieve such a storage requiring merely 1.43 megabytes of storage. Whilst both such space requirements are easily satisfied by modern handheld devices, as the average smartphone possesses over 4GB of RAM (GSMArena, 2018), the run-time benefits do introduce a noteworthy improvement. Figure 8 showcases the speed-up provided by Bloom filters. The Bloom filter in questions uses the following parameters: **[TODO:** add params]. On a dataset of 100,000 revoked hashes, one can see that, as expected, the runtimes increase linearly. The x-axis varies the percentage of the candidates which are an actual member of the test data set. In other words, the percentage of actual matches increases in each subsequent measure. As expected, the verification utilising solely a Bloom filter is not impacted by this variation. Similarly, verification solely utilising Binary Search is also relatively unimpacted. The variation only utilising binary search on a possible match in the Bloom filter, is impacted the most. This variant only makes (expensive) I/O operations when the Bloom filter reports a possible match. As becomes apparent, the benefits from the Bloom filter decrease with the increase of the membership percentage. Hence, the speed-up is most prominent with lower membership percentage. In terms of attestation verification, a Bloom filter is thus most beneficent in case the vast majority of the encountered attestations are non-revoked and, thus, valid. We draw the conclusion based on the reported statistic by Nieuwsuur (2019), which stated that in 2018, in the Netherlands nearly 340.000 official identification documentation was lost. Percentage-wise, this leads to an annual 2% loss based on the 17.18 million residents of that year (CBS, n.d.). Note that this estimation does not include the number of different identification documents hold by a resident (e.g. driving license, passport, and identification card). As a consequence this actual number most likely differs greatly. Also note that the properties of physical identification measures do not necessary directly translate to any digital variants, as, most desirably, digital credentials are far more difficult to lose. However, this showcases that it can be expected to encounter far more valid attestations than revoked ones. Especially with the assumption that the majority of the network is honest. To conclude, we deem the speed-up benefits provided by the usage of Bloom filters to be significant.

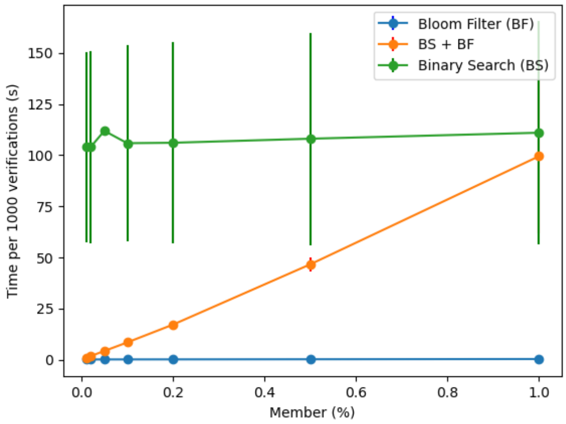

Figure 8

Verification runtime per 1000 transactions (n=100,000

Communication

In order to save resources on clients, a client will only connect to a set amount of peers at any time. For obvious reasons, connecting to every other client would lead to a tremendous amount of overhead and resources used. As such, a client will connect to arbitrary clients as per IPv8 specification. In order to force visibility of certain peers, we propose the usage of secure multi-party communication channels. These channels are to be constructed through the usage of passphrases, which are possibly, but not necessarily, nonces. Only clients possessing the specific phrase, are able to connect to the channel, however, as may become apparent, such channels are still vulnerable to eavesdroppers. As such, encryption must still be used, however, the communication with the desired peers is guaranteed.

Related Works

Revocation

As a key contribution of our work lays in the field of revocation, we compare our works with the current state of the art in literature. We note that literature on revocation in Self-Sovereign Identity systems is not a widely discussed topic in academia, as such, the selected articles discuss revocation on a broader scale of digital identities. Table 2 displays the high level comparison of HRM compared to related revocation algorithms. The several comparison characteristics are quantified on a Low/Medium/High scale, where the actual perfor-

Table 2

Revocation comparison with related works

	HRM	**Lasla et al. (2018)[1]**	**Popescu et al. (2003)**	**Liau et al. (2005)**	**Haas et al. (2011)[2]**
Offline availability	✓	✓	✗	✓	✓
Authority interactivity	✗	✗	✓	✓	✗
Storage requirement	**Med**	High	Low	High	Med
Processing requirement	**Low**	High	High	Low	Low
Revocation latency	**Low**	High	Low	Low	Med
Verification latency	**Low**	Low	Med	High	Low
SPOF	✗	✗	✓	✗	✗
False positives	✗	✗	✗	✗	✓
False negatives	✗	✗	✓	✗	✗

[1] As no specification on the type of blockchain was given, we assume the usage of the Bitcoin blockchain as per their test results.
[2] As the propagation of revocations is possibly dependent on both the distribution through Vehicle-to-Vehicle communication and Road-Side Units, we assume a revocation latency higher than direct client-to-client communication and lower than that of a (PoW) blockchain.

mances are broad estimations based on the used technologies (e.g. blockchain). We note the selected related works either discuss more generic node revocation (Liau et al., 2005; Popescu et al., 2003) or are particularly tailored to Vehicular ad-hoc networks (VANETs) (Lasla et al., 2018; Haas et al., 2011). However, the general concept discussed in these works is the revocation of certificates, which could relatively trivially be transformed into the Self-Sovereign Identity domain as the expected loads of these systems can be shown to be compatible. For instance, Haas et al. (2011) assumes up to 25 million revocations, which is less than our assumed quantities.

As becomes apparent from Table 1, HRM is expected to outperform previously proposed solutions on most characteristics. We note that HRM has a relatively higher storage requirement due to the storage of the raw revocations. However, as discussed previously, the storage can be narrowed down to a few megabytes through solely using a Bloom filter as opposed to additionally using cold storage. This, ofcourse, comes with the drawback of false positives and, as such, depends on the use-case of the client.

Conclusion

We presented a Self-Sovereign Identity framework which can facilitate the digital identity needs of the European Union and can be deemed to be of *Industry Grade*. IG-SSI includes the possibility of attestation signing, presentation, verification and most notably, revocation. The scheme is shown to work fully distributed through the usage of IPv8 and allows for fully distributed revocation. Privacy is aided through the usage of zero-knowledge proofs and secure communication with specific peers can be forced through passphrases. A reference implementation for the semantic layer has been created, as well as a mobile client showcasing full usability on smartphones. Legally valid signatures can be achieved through IG-SSI, however, usability of the scheme has to be generated through global adoptation. As the scheme has been developed in collaboration with the Dutch Ministry of the Interior and Kingdom Relations

References

Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, *13*(7), 422–426.

Boneh, D., Goh, E. J., & Nissim, K. (2005). Evaluating 2-DNF formulas on ciphertexts. In *Lecture notes in computer science* (Vol. 3378, pp. 325–341). Springer Verlag. Retrieved from `https://link-springer-com.tudelft.idm.oclc.org/chapter/10.1007/978-3-540-30576-7_18` doi: 10.1007/978-3-540-30576-7{_}18

Boudot, F. (2000). Efficient Proofs that a Committed Number Lies in an Interval. In B. Preneel (Ed.), *Advances in cryptology — eurocrypt 2000* (pp. 431–444). Berlin, Heidelberg: Springer Berlin Heidelberg.

Buterin, V. (2013). A next generation smart contract & decentralized application platform. *Ethereum Foundation.*

Cameron, K. (2005). The laws of identity. *Microsoft Corp*, *5*, 8–11.

CBS. (n.d.). *Bevolkingsteller.* Retrieved from `https://www.cbs.nl/nl-nl/visualisaties/dashboard-bevolking/bevolkingsteller`

GSMArena. (2018, 4). *Counterclockwise: RAM capacity through the years* . Retrieved from `https://`

www.gsmarena.com/counterclockwise_ram_capacity_through_the_years-news-30756.php

Haas, J. J., Hu, Y. C., & Laberteaux, K. P. (2011, 3). Efficient certificate revocation list organization and distribution. *IEEE Journal on Selected Areas in Communications*, *29*(3), 595–604. doi: 10.1109/JSAC.2011.110309

Halkes, G., & Pouwelse, J. (2011). UDP NAT and firewall puncturing in the wild. In *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)* (Vol. 6641 LNCS, pp. 1–12). Springer, Berlin, Heidelberg. Retrieved from https://link-springer-com.tudelft.idm.oclc.org/chapter/10.1007/978-3-642-20798-3_1 doi: 10.1007/978-3-642-20798-3{_}1

Khovratovich, D., & Law, J. (2017). *Sovrin: digital identities in the blockchain era* (Tech. Rep.). Retrieved from http://www.credentica.com/the

Koens, T., Ramaekers, C., & Van Wijk, C. (2018). *Efficient Zero-Knowledge Range Proofs in Ethereum* (Tech. Rep.). ING.

Lasla, N., Younis, M., Znaidi, W., & Ben Arbia, D. (2018, 3). Efficient Distributed Admission and Revocation Using Blockchain for Cooperative ITS. In *2018 9th ifip international conference on new technologies, mobility and security, ntms 2018 - proceedings* (Vol. 2018-January, pp. 1–5). Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/NTMS.2018.8328734

Liau, C. Y., Bressan, S., & Tan, K.-L. (2005). Efficient Certificate Revocation : A P2P Approach. *HICSS'05*.

Nakamoto, S. (2009). *Bitcoin: A Peer-to-Peer Electronic Cash System* (Tech. Rep.). Retrieved from www.bitcoin.org

Nieuwsuur. (2019, 7). *We verliezen massaal onze paspoorten: fraudemeldingen verdubbeld | Nieuwsuur.* Retrieved from https://nos.nl/nieuwsuur/artikel/2292728-we-verliezen-massaal-onze-paspoorten-fraudemeldingen-verdubbeld

Peng, K., & Bao, F. (2010). An efficient range proof scheme. In *Proceedings - socialcom 2010: 2nd ieee international conference on social computing, passat 2010: 2nd ieee international conference on privacy, security, risk and trust* (pp. 826–833). doi: 10.1109/SocialCom.2010.125

Popescu, B. C., Crispo, B., & Tanenbaum, A. S. (2003). A certificate revocation scheme for a large-scale highly replicated distributed system. In *Proceedings - ieee symposium on computers and communications* (pp. 225–231). doi: 10.1109/ISCC.2003.1214126

Raya, M., Papadimitratos, P., Aad, I., Jungels, D., & Hubaux, J.-P. (2007). Eviction of Misbehaving and Faulty Nodes in Vehicular Networks. *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, *25*(8). Retrieved from http://www.sevecom.org doi: 10.1109/JSAC.2007.0710xx

Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, *21*(2), 120–126.

Smart, N. P. (2016). In *Cryptography made simple* (p. 425–438). Springer.

Stigler, G. J. (1964). A theory of oligopoly. *Journal of Political Economy*, *72*(1), 44–61. Retrieved from http://www.jstor.org/stable/1828791

Stokkink, Q., Epema, D., & Pouwelse, J. (2020). A Truly Self-Sovereign Identity System. *arXiv preprint arXiv:2007.00415.*

Stokkink, Q., & Pouwelse, J. (2018). Deployment of a blockchain-based self-sovereign identity. In *2018 ieee international conference on internet of things (ithings) and ieee green computing and communications (greencom) and ieee cyber, physical and social computing (cpscom) and ieee smart data (smartdata)* (pp. 1336–1342).

Von der Leyen, U. (2020, 9 16). *State of the union address by president von der leyen at the european parliament plenary.* Retrieved from https://ec.europa.eu/commission/presscorner/detail/en/SPEECH_20_1655

Zeilemaker, N., Schoon, B., & Pouwelse, J. (2013). *Dispersy bundle synchronization* (Tech. Rep. No. PDS-2013-002). TU Delft. Retrieved from http://www.pds.ewi.tudelft.nl/fileadmin/pds/reports/2013/PDS-2013-002.pdf

Zhou, T., Li, X., & Zhao, H. (2019). EverSSDI: Blockchain-based framework for verification, authorisation and recovery of self-sovereign identity using smart contracts. *International Journal of Computer Applications in Technology*, *60*(3), 281–295. doi: 10.1504/IJCAT.2019.100300