Distributed Attestation Revocation in Self-Sovereign Identity [DRAFT]

R.M. Chotkan and J.A. Pouwelse
R.M.Chotkan@student.tudelft.nl, J.A.Pouwelse@tudelft.nl

***Abstract*—The internet was created without a standardised identity layer, resulting in the management of a plethora of digital identities which hold no legal value. Moreover, often requiring cumbersome identity card checks, e.g., through digital photocopies. Initiatives such as *User-centring identities* have mostly failed, resulting in asymmetrical control held by Big Tech in digital identities. Self-Sovereign Identity (SSI) can prove to overcome these hurdles. SSI aims to put one at the centre of their digital presence, enabling ownership over one's digital identity. Furthermore, opening up the possibility for legally valid digital identities. Our research addresses the key issue of revocation of SSI credentials. Revocation is hampering the up-rise of the SSI concept: existing attempts critically rely on communication with central authorities and introduce inequalities into the architecture. We present a fully distributed SSI framework with the first fully distributed SSI revocation mechanism requiring no specialised nodes, in which equality and offline usability are at the core of the architecture. A novel gossip-based revocation algorithm propagates revocations throughout the network, enabling offline verification. Furthermore, the resulting framework allows for attestation signing, presentation and verification in Zero-Knowledge. Our result show improvements with respect to the state of the art. We claim that our architecture is a viable candidate for the upcoming European-wide identity standard. Our small-scale trial shows that this is a promising direction to further explore.**

I. Introduction

Since the onset of the Information age, digital trust has been an issue requiring many workarounds. The core concepts of the internet are 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). The popularity of identity management solutions by Big Tech has resulted in an oligopoly in digital identity (Siftery, 2017). Wherein a regular oligopoly consumers are at a price-wise disadvantage (Stigler, 1964), in this technical oligopoly the identity providers have an asymmetrical control of ones digital presence. Furthermore, increasing needs for digital identities from governments such as the European Union (Von der Leyen, 2020), has portrayed to need for and relevancy of the field. This is further fuelled by the urgency of COVID-19 vaccination passports (European Commission, 2021), requiring digital validity across borders.

The *Self-Sovereign Identity* (SSI) concept can prove to fill this digital and societal gap. SSI aims to generate digital trust by providing verifiable digital identities, putting the user at the centre. SSI is a concept requiring cutting-edge concepts such as decentralised ledger (DL) technology and decentralised public key infrastructure (DKPI). A key issue in SSI remains the *revocation* of issued credentials. As portrayed by Table I, distributed revocation is to our knowledge yet to be solved in SSI. Existing SSI solutions such as Sovrin[1], Serto[2] and Irma[3] solve the issue of revocation through specialised verification nodes. This disallows offline verification and introduce inherent inequalities in the network, possibly leading to censorship or collusion (Khovratovich & Law, 2017).

This research introduces an academic Self-Sovereign Identity framework focusing on distributed revocation, offline verification, and intrinsic equality across the network. The scheme is based on the previous works by Stokkink & Pouwelse (2018); Stokkink et al. (2020). The following contributions are made: (1) the first fully distributed revocation algorithm for SSI, achieving reliable revocation over unreliable communication links and (2) offline verification of verifiable claims (VCs). Furthermore, a reference implementation of the semantic layer is created using the IPv8 protocol stack (Halkes & Pouwelse, 2011; Zeilemaker et al., 2013) as well as a proof-of-concept application portraying the feasibility of SSI and distributed revocation on handheld devices. An implementation of the framework has been validated in a small-scale trial.

II. Problem Description

Revocation is required in the instance that a credential becomes prematurely voided. Revocations must be made apparent to the parties for whom it is possible to encounter the corresponding credential. Verification of revoked credentials must lead to failure. As any client may be in Authority in an SSI system and revocations must be reachable by any client, the propagation of revocations must be performed in such a fashion that confidentiality, integrity, and availability are ensured.

Revocation mechanisms are present in traditional Public Key Infrastructures (PKIs) such as PKIX (IETF, n.d.). Broadly speaking, a PKI uses a Certificate Authority (CA) to publish a Certificate Revocation List (CRL), containing revoked certificates. In this structure, CAs are inherently central authorities, having relatively absolute power over revocations.

This issue complicates in the SSI domain as any client may be an Authority. Transforming the PKI structure to SSI would lead—trivially—to each client contacting all authorities on

[1]For Sovrin, see: `https://sovrin.org/`
[2]For Serto, see: `https://www.serto.id/`
[3]For Irma, see: `https://irma.app/?lang=en`

TABLE I: Revocation comparison with related works

	Domain	Type	Means	Description	No network operators	Offline availability	No Authority interactivity	Offline Verification	No SPOF	No FPs/FNs
HRM (this work)	SSI	Attestation	Hash	Gossip-based p2p propagation of revocations.	✓	✓	✓	✓	✓	✓
Xu et al. (2020)	SSI	Node	PK	List of accepted nodes stored on blockchain.	✗	✓	✓	✗	✓	✓
Abraham et al. (2020)	SSI	Attestation	Hash	Revocations stored on blockchain	✓[1]	✓	✓	✓	✓	✓
Lasla et al. (2018)	C-ITS	Node	Hash	Revocations stored blockchain and RSUs.	✓[1]	✓	✓	✗	✓	✓
Popescu et al. (2003)	DS	Certificate	CRL	Revocations handled locally by Authority.	✓	✗	✗	✓	✗	✗
Liau et al. (2005)	P2P	Certificate	CRL	Uses distribution points and P2P communication.	✗	✓	✗	✓	✓	✓
Haas et al. (2011)	VANET	Certificate	CRL	RSUs and v2v propagation.	✗	✓	✓	✓	✓	✗
Laberteaux et al. (2008)	VANET	Certificate	CRL	RSUs and v2v propagation.	✗	✓	✓	✓	✓	✓
Eschenauer & Gligor (2002)	DSN	Node	PK	Single Authority propagates revocation of nodes.	✗	✓	✗	✓	✗	✓
IRMA	SSI	Attestation	C.A.	Revocations stored on permissioned blockchain.	✗	✗	✓	✗	✓	✗
SOVRIN	SSI	Attestation	C.A.	Revocations stored on permission blockchain.	✗	✗	✓	✗	✓	✓
uPORT	SSI	Attestation	Hash	Revocations stored on public blockchain.	✓	✗	✓	✗	✓	✓

[1] As no specification on the type of blockchain was given, we assume the usage of a public permissionless blockchain.

interval to receive revocations. This may introduce single point of failures in the mechanism or requires much infrastructure for distribution. Furthermore, the ever increasing size of a CRL-esque structure leads to much overhead.

Deployed SSI solutions mostly introduce specialised authorities, e.g. in Sovrin (n.d.); by Design Foundation (n.d.), or expensive Proof-of-Work blockchains (uPort, n.d.) for maintaining the network. Such authorities may introduce privacy issues, collusion (Khovratovich & Law, 2017), or censorship. Apart from these issues, most revocation mechanisms are dependent on *cryptographic accumulators* (Hardman, 2019, 2018; IRMA, n.d.). Whilst cryptographic accumulators are privacy-preserving, they disallow offline validation due to requiring witness updates. As such, both the Subject and the Verifier must be fully updated during presentation-time. Furthermore, cryptographic accumulators are computationally expensive to such an extend that they are discouraged to be used at each verification (IRMA, n.d.).

In academia, proposed solutions include the usage of blockchain as a storing structure (Zhou et al., 2019) or require active checks in the worst case during verification (Stokkink & Pouwelse, 2018) (see more in section III).

Based on the previous shortcomings, we believe that the lack of revocation is hampering the up-rise of SSI. This research proposes the first fully distributed revocation algorithm for SSI, enabling offline verifiability whilst relaxing requirements which are critical in prior works, such as central authorities and reliable Internet. The revocation mechanism is part of an SSI scheme which requires no specialised nodes, leading to a first fully distributed SSI scheme with offline verification and intrinsic equality.

III. Related Work

As the key contribution addresses revocation, we focus on related work discussing this topic. We note that literature on revocation in Self-Sovereign Identity systems is not a widely discussed topic in academia, as such, the selected works address distributed revocation on a broader scale. We group related works in the revocations of SSI credentials, certificates, and nodes.

Revocation of SSI Credentials

Hardman (2018, 2019); IRMA (n.d.) propose the usage of cryptographic accumulators for revocation in SSI. This requires cliens to update their witness through Authorities in order to proof non-revocability of credentials. Xu et al. (2020) uses a blockchain for storing legitimate clients, indirectly disallowing access for revoked clients in the SSI system. Updating the client list is performed by network operators, which can be deemed Authorities. Abraham et al. (2020) propose the usage of a blockchain to store revoked signatures, on which consensus is reached through the nodes of the network.

Revocation of Certificates

Laberteaux et al. (2008) discuss the revocation of certificates in Vehicular ad hoc networks (VANETS) through distribution of CRLs. Distribution is handled through Road Side Units (RSUs), which are specialised propagation nodes and through epidemic spread between vehicles. Haas et al. (2011) build upon this work by showcasing the practicality using differentiating deployment rates and guaranteeing a certain degree of privacy. Liau et al. (2005) propose the distribution of CRLs through direct peer updates, reducing the communication overhead caused by periodic CRL updates, signatures over CRLs allow nodes to built trust in others. Popescu et al. (2003) discuss revoking certificates based on the clustering of clients and probabilistic auditing for honesty.

Revocation of Nodes

Eschenauer & Gligor (2002) discuss the revocation of nodes in distributed sensor networks. Revocation is handled by a specialised authority, delegating revocation orders to regular sensor clients. Lasla et al. (2018) discuss the revocation of malicious vehicles in Cooperative Intelligent Transportation Systems (CITS). Their solutions uses a blockchain for storing revocations through a distributed vehicle admission and revocation scheme.

Conclusion

Table I portrays that, to our knowledge, no fully distributed SSI revocation mechanism has been proposed, apart from those utilising blockchains. We note that existing blockchains suffer from the reliance on reliable Internet and the downloading and verification of blocks, hindering offline verifiability and introducing overhead. As such, this research proposes the first fully distributed revocation algorithm for SSI, enabling offline verification of verifiable claims. Furthermore, improving the state-of-the-art in revocation in SSI systems.

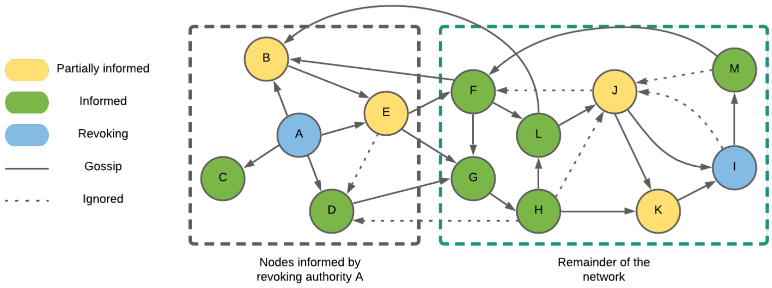


Fig. 1: Revocation gossip in a network

IV. Architecture & Analysis

Specialised verification nodes for managing revocations, present in e.g. Zhou et al. (2019); Tobin & Reed (2016); by Design Foundation (n.d.), deteriorate equality in the network and could even lead to censorship or collusion (Khovratovich & Law, 2017). As our proposed architecture is designed without such nodes, the trivial solution for revocation is to actively query the Authorities in order to verify that they still attest for a claim (Stokkink & Pouwelse, 2018). This querying requires interactivity with the Authorities, disallowing offline verification. Whilst availability often is a key characteristic in distributed systems, there is no guarantee that specific clients are available. As our architecture allows for an indefinite amount of attestations for a single Claim, interactivity with the Authorities can prove to become rather unmanageable as it introduces additional verification time due to additional network traffic and response times of Authorities.

Our novel revocation mechanism (Figure 1) attempts to overcome the hurdle of interactivity whilst allowing for offline verification. During verification-time, clients do not require to be online, they merely require occasional synchronisation of revoked attestations through communication with other peers. Next, we discuss the three main components of the design.

A. *Trusted Authorities*

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. 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. We propose the usage of a Trusted Authority Storage (TAS). In the TAS, the public keys of Trusted Authorities (TAs) are stored. The distinction between acknowledged (trusted) and Unacknowledged Authorities (UAs) is made. 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 the revocations of TAs. The results of acceptance are the storage of the revoked signatures and propagation towards network.

B. *Offline Revocation List*

Any valid received revocation should be stored by a client for later reference in 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, 2006) 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.

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.

C. *Propagation*

The propagation of revocations requires a protocol that ensures information is 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 a gossip protocol with interval re-transmissions. Gossip protocols are communication protocols which allow for the periodic exchange of data with (random) peers (Kwiatkowska et al., 2008). They are originally modelled after epidemic spread (Demers et al., 1987).

The gossip between clients has been visualised in Figure 1, portraying the communication of revocations from an initial authority to a select subset of clients. Clients can built trust in the revocations through signatures provided by the revoking authority. After which these clients gossip the revocation to the remainder of the network. The gossip between clients is further chronologically ordered in Figure 2, in which it can be seen that clients may only be partially aware of revocations at a certain instance and only eventually receiving the remainder (e.g., see node F). This is due to the usage of a multi-step update procedure, in order to decrease the overhead of gossiping a theoretically unbound number of signatures.

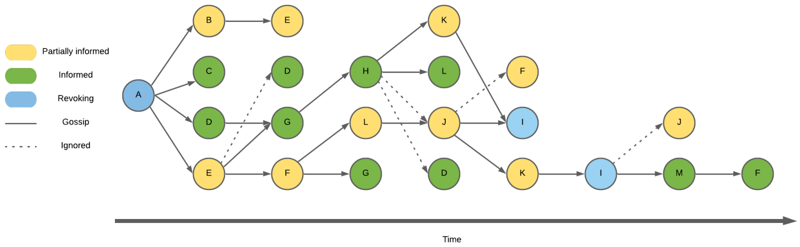


Fig. 2: Revocation gossip over time

The message flow between a gossiping and receiving client, following from the multi-step procedure, is visualised in Figure 3. The gossip is split-up in two phases: firstly, a gossiping client gives notice to another client that it possesses specific revocations. 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 under-bound of the known versions apparent. Finally, the gossiping client sends the revocations to the updating client. This additional step loosens network requirements for receiving clients. Clients may become spontaneously online or go sporadically offline, resulting in missing revocations (as is also modelled in Figure 1 and 2). As such, this mechanism allows partial updates. Furthermore, overhead is further reduces as clients are not interested in revocations belonging to non acknowledged authorities or a client may already be aware of all revocations.

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. Furthermore as opposed to selecting an under-bound on revocation versions, a client may request specific versions in order to reduce network usage. Furthermore, we note that this procedure can be fine-tuned by only propagating the Bloom filter contents as proposed by (Haas et al., 2011)

D. Theoretical Analysis

We consider a network of distributed agents denoted by a complete graph $G = (V, E, w)$. Where V is the set of agents, E the set of edges between agents, and w representing the delays between nodes. An edge $(i, j) \in E$ represents a throughput link of information from node i to j. Agents do not necessarily have full knowledge of G, but do have knowledge on a subset of G, representative as neighbours.

The propagation of the revocations is dependent on both delays imposed by the protocol and by the network. For protocol delays, the propagation time is dependent on the parameters imposed on the protocol, being:

- **Gossip-interval** (t_g): the time interval on which peers are gossiped to.
- **Gossip amount** (n_g): the number of peers which are gossiped to on a time interval.
- **Peer selection** ($\mathcal{F}_g(x)$): the function used to determine which peers are gossiped to.

Definition 4.1: (Protocol delays). Let n_p be the size of G and let $g = t_g \cdot \dfrac{n_p}{n_g}$ be the minimal number of interval iterations required to gossip to all peers. The peer selection function $\mathcal{F}_g(X)$ may result in overlapping subsets. I.e., let $f_i = F_g(P)$ be the subset of peers generated at iteration i and let $f_{i+j} = F_g(P)$ be the subset generated at iteration $i + j$, then it does not necessarily hold that $f_i \cap f_{i+j} = \emptyset$. Hence, let $P_f = p_0, ..., p_{n-1}$ be the multiset of peers of size $m_p >= n_p$ selected throughout each iteration until convergence. I.e., the peer selection function $\mathcal{F}_g(X)$ selected at least $m_p >= n_p$ peers, leading to at least $t_g \cdot \dfrac{m_p}{n_g}$ iterations. The additional iterations can be modelled by: $h = t_g \cdot \dfrac{m_p - n_p}{n_g}$, where $h \geq g$. This leads to the propagation time for the protocol delays for a single client i attempting to gossip a single update to the entire visible network with size n as to be as summarised

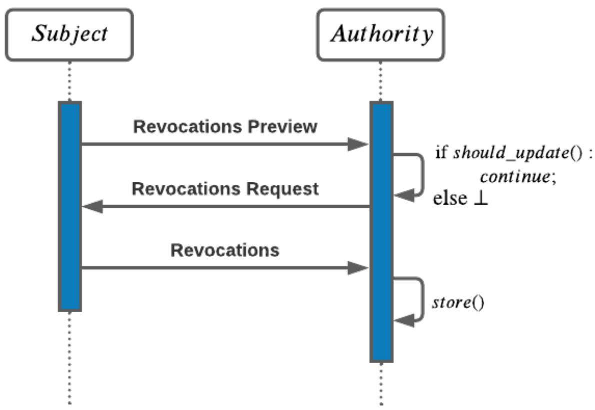


Fig. 3: Multi-step Update Procedure

in Equation 1.

$$
\begin{aligned}
\mathcal{T}_{p_i} &= h + g \\
&= t_g \cdot \frac{n_p}{n_g} + t_g \cdot \frac{m_p - n_p}{n_g} \\
&= t_g \cdot \left(\frac{n_p}{n_g} + \frac{m_p - n_p}{n_g}\right) \\
&= t_g \cdot \frac{m_p}{n_g}
\end{aligned} \tag{1}
$$

This can be generalised for the entirety of the network as visible in Equation 1. As clients are not aware of their position in the network (relatively to others) or of the peers already contacted by other clients, there can only be set an upper bound on the expected runtime of the algorithm, as each peer attempts to gossip all information to all other peers. Hence, the propagation delay can be summarised to the formula presented in Equation 2, where $t_{g_i}, m_{p_i}, n_{g_i}$ are the gossip-interval, the maximum number of gossiped peers, and gossip amount per iteration for client i, respectively.

$$
\begin{aligned}
\mathcal{T}_p &\leq \sum_{i=0}^{n-1} \mathcal{T}_{p,i} \\
&\leq \sum_{i=0}^{n-1} \left(t_{g_i} \cdot \frac{m_{p_i}}{n_{g_i}}\right)
\end{aligned} \tag{2}
$$

Definition 4.2: (Network delays). Next, we generalise the delays imposed by the network. Let $\delta_{i,j}$ be the propagation delay from node i to node j and let function $\Delta(p_j)$ compute the smallest propagation delay for node p_j to be gossiped to. I.e., $\forall(p_i, p_k) \in \{p_0, ..., p_{n-1}\}$ it holds that $\delta_{i,j} < \delta_{k,j}$. Finally, let $\mathcal{C} = \{c_0, \ldots, c_{n-1}\}$ be the the set of delays imposed by processing times on the clients on invocation $\Delta(p_j)$. This leads to the network delay for a single client i updating the entirety of the to him visible network with size n as summarised in Equation 3

$$
\mathcal{T}_{n_i} = \sum_{j=0}^{n-1} (\delta_{i,j} + c_j) \tag{3}
$$

Then, the total network delays in a system with a set of $P = \{p_0, \ldots, p_{n-1}\}$ nodes of size n can be modelled as visible in Equation 4.

$$
\mathcal{T}_n = \sum_{i=0}^{n-1} (\Delta(p_i) + c_i) \tag{4}
$$

Definition 4.3: (Propagation time). Definition 4.1 and Definition 4.2 lead to a total propagation time as visible in Equation 5. Again, due to the distributed nature and possible randomness of peer selection, only an upper-bound can be assigned.

$$
\begin{aligned}
\mathcal{T} &= \mathcal{T}_p + \mathcal{T}_n \\
&\leq \left(\sum_{i=0}^{n-1} \left(t_{g_i} \cdot \frac{m_{p_i}}{n_{g_i}}\right)\right) + \left(\sum_{i=0}^{n-1} \Delta(p_i) + c_i\right) \\
&\leq \sum_{i=0}^{n-1} \left(t_{g_i} \cdot \frac{m_{p_i}}{n_{g_i}} + \Delta(p_i) + c_i\right)
\end{aligned} \tag{5}
$$

Equation 5 leads to a runtime of $\mathcal{O}(n)$, as in the worst case a single client updates all other clients. However, it is expected to be logarithmic with respect to the number of nodes, as each gossiped to node can gossip to yet uninformed nodes. More specifically, a node n_i can gossip to node n_j whilst node n_k gossips to node n_l. As such, the more nodes become informed, the faster the remaining nodes are gossiped to.

Theorem 4.1: Each client will eventually receive all revocations. Clients may be sporadically online and still receive all revocations, albeit possibly in non-consecutive order, without affecting availability of reachable clients.
We defined the minimum number of iterations ($t_g \cdot \frac{m_p}{n_g}$) to be dependent on the used client selection algorithm (see Definition 4.1). Clients are unreliable as they may be sporadically online due to unreliable connections. As revocations are gossiped on an interval to clients, regardless of whether they have been reached prior, it is infeasible that overlapping subsets of selection by $\mathcal{F}_g(x)$ are not created by a random number generator (RNG) (as the chances of each peer being selected grow to 100%). As a consequence any failed gossip attempt or temporary offline clients will be gossiped to by any other client at a later instance. Especially since each (honest) client attempts to reach each other client. Furthermore, as revocations are split into different sets a sporadically online client may receive version $i+j$ prior to version i, however, as it has been shown that a client will be reached again, i will be received at a later instance. This leads to each client eventually receiving all revocations, regardless of reliability of connections.

Theorem 4.2: Revocations in any network with at least 1 honest node will propagate to each client in $\mathcal{O}(n)$ in at most $\sum_{i=0}^{n-1}\left(t_{g_i} \cdot \frac{m_{p_i}}{n_{g_i}} + \Delta(p_i) + c_i\right)$ seconds.
Consider a network with n nodes, of which m nodes are not aware of the latest revocations. Of the $n - m$ nodes, which are aware of the latest revocations, all but one node c_i is malicious. We assume that dishonest nodes cannot affect network traffic. Deteriorating the condition that the network is comprised of a complete graph, we assume that the honest node eventually has connectivity with at least a single node c_j belonging to the m ungossiped nodes. Using Theorem 4.1 we conclude that c_i is able to eventually gossip revocations to c_j. Subsequently, c_j eventually gossips—albeit possibly indirectly— to the remainder of the group of uninformed nodes. As such, we conclude that revocations propagate across a network in case there exists at least a single honest node.

In the worst case, c_i gossips to each node belonging to the m nodes, resulting in a runtime of $O(n)$ and a propagation time of $\sum_{i=0}^{n-1}\left(t_{g_i}\cdot\frac{m_{p_i}}{n_{g_i}}+\Delta(p_i)+c_i\right)$. Note that this node may be the Authority of the revocations.

E. Attestation Interactions

Self-Sovereign Identity is built around *Verifiable Claims* (VCs) (Mühle et al., 2018), which are composed of several types of information. Firstly, a *claim* is made by a Subject (Sporny et al., 2019). Authorities can attest to a claim, making it a VC. When metadata is added to a VC, we speak of an Attribute. Finally, a set of related attributes is referred to as a *Credential* (Mühle et al., 2018).

In the proposed design, each *Claim* is represented by an anonymised *Token*, which stores a reference to a claim via its hash. A *Token* can be references by multiple *Metadata* structures which assign different properties to a *Claim* (e.g. a validity term). Furthermore, multiple *Attestations* can be made for a *Metadata*. Finally, although not explicitly modelled, multiple *Credentials* can reference multiple *Attestations* and as such, multiple *Claims*. The *Tokens* are stored in a Blockchain-esque structure, referencing the previous Token. This aids in preventing the withholdment of claims as well as making it more difficult for one to use stolen credentials. The first token, comparable to a genesis-block in Blockchain structures such as Nakamoto (2009), contains the hash of the public key of the Subject. Any subsequent Credential, thus, generates a new Token, occupying a place as a shackle in the chain. 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).

Next, we discuss the lifecycle of these credentials. We identify three main interactions surrounding VCs:

1) Attestation Signing
2) Attribute Presentation
3) Attribute Verification

F. Attestation Signing

The attestation procedure is visible in Figure 4. It consists of two phases: the Claim-phase and the Attestation-phase which do not necessarily require subsequent execution. More specifically, for a single to be attested claim, the Claim-phase requires a single execution, which must occur before the Attestation-phase. Whilst subsequently, the Attestation-phase can be performed indefinitely by different Authorities.

1) Claim-phase: The Claim-phase is initiated by a Subject through a request. In this request a subject makes metadata such as its public key, the proof format and the attribute name apparent. The public key belongs to a single-use key pair, strengthening privacy. The Authority may respond by creating a Zero-Knowledge Proof incorporating the value belonging to the requested claim. As may become apparent from this description two modus operandi are possible. Firstly, a client may self-create this claim, following the natural description of a claim. However, a client may not know the associated claim.

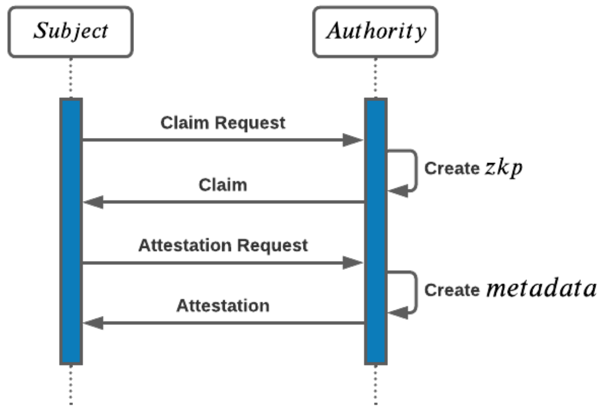


Fig. 4: Attestation Flow

Hence, the second modus operandi delegates the creation of the claim to an Authority, not requiring prior sharing of the claim value.

2) Attestation-phase: After possessing a claim, a Subject request an attestation for said Claim, creating a VC and subsequently an Attribute. When a Subject requests an attestation from Authority it discloses the prior attestations and tokens, allowing the Authority to verify previous attributes. Furthermore, the Authority creates metadata for properties of the attestation, including the hash of the plaintext value. The attestation is made through a signature over the claim. However, as a hash would allow for trivial preimage attacks for attributes with a limited message space (e.g. an *age* credential), we propose the usage of salts (Arias, 2021).

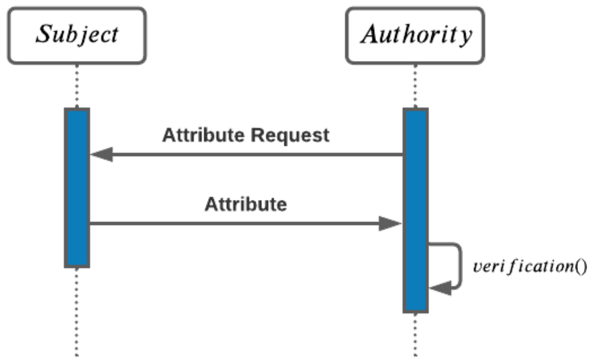


Fig. 5: Attestation Presentation

G. Presentation Flow

The interactions for the presentation of attributes is portrayed in Figure 5. 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. Next, similarly to the Attestation flow, an Authority may request the tokens of previous claims until has gain enough confidence. Note here that the attribute 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.

H. Verification Flow

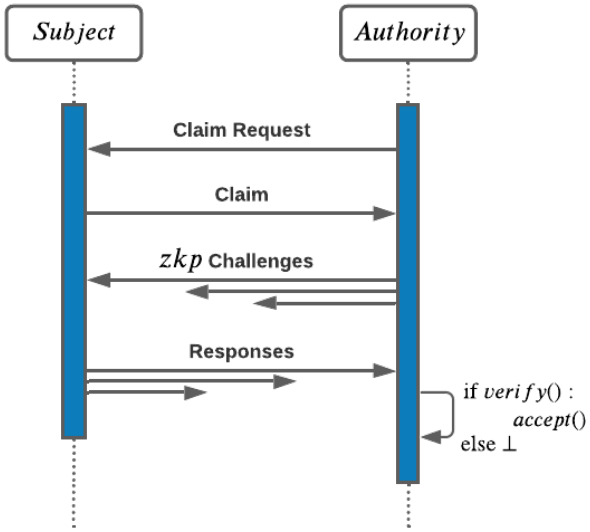


Fig. 6: Interactive Verification

We propose two types of verification: an interactive and a non-interactive variant. Both methodologies use the attestations made by authorities. Hence, the list of attestors must contain an Authority that is trusted by the Verifier.

The former variant is presented in Figure 6. For active verification, a Verifier requests the underlying Claim of an attribute from the Subject (procured through prior presentation). The Subject may consent by sending the requested Claim. Next the Verifier may send challenges to verify the underlying ZKP. 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. Furthermore, the Authority verifies the presented attestations.

The second method for verification solely uses the attestations. 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. it can be performed through QR-codes), allowing full offline verification. 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.

V. Algorithms & Simulation

The analysis of the revocation mechanism is two-fold. Firstly, we discuss a simulation showcasing scalability amongst high numbers of clients (up to 10.000). Secondly, we showcase analysis through deployment of smartphones in section VII, portaying the usability on mobile clients. The simulations were performed on a system with a i7-6700HQ clocked at 2.60 GHz and 16GB of RAM.

Algorithm 1: Revocation Gossip

input : Set of Clients in the network $\mathcal{C} = \{c_0, \ldots, c_i\}$, Set of known Authority-Version pairs $\mathcal{A} = \{(a_0, v_j), \ldots, (a_j, v_k)\}$ Gossip interval t_g, Peer selection amount n_g
output: Revocation update gossip

```
while True do
    C_g ← SelectPeers (C, n_g);
    foreach c_i ∈ C_g do
        GossipRevocations(c_i, A);
    Wait(t_g);
```

Algorithm 2: Revocation Update Request Procedure

input : Set of Authority-Version pairs $\mathcal{A} = \{(a_0, v_j), \ldots, (a_j, v_k)\}$, Set of trusted Authorities (TAS) $\mathcal{T} = \{t_0, \ldots, t_n\}$
output: Revocation update request

```
On reception of A by Client c_i;
for Authority a_i, Version v_j in A do
    if r_i ∉ ORL then
        v_local ← FindMissingVersion(a_i);
        if v_local < v_j then
            RequestUpdate(c_i,a_i,v_j);
```

A. Simulation

The simulation was performed through mimicking the gossip of 1 million revocations between clients. Each client runs three algorithms. A gossiping client runs algorithm 1, after which a receiving client initiates algorithm 2. Finally, the revocations are send as modelled by algorithm 3. As the simulation is performed on a single machine, network usage was of no impact. As such, arbitrary delays between 20-50 ms are introduced in order to simulate the impact of network

Algorithm 3: Revocation Gossip

input : Set of Clients in the network $\mathcal{C} = \{c_0, \dots, c_i\}$, Set of known Authority-Version pairs $\mathcal{A} = \{(a_0, v_j), \dots, (a_j, v_k)\}$ Gossip interval t_g, Peer selection amount n_g

output: Revocation update gossip

```
for Authority a_i, Version v_j, , in A do
    C_g ← SelectPeers (C, n_g);
    foreach c_i ∈ C_g do
        GossipRevocations(c_i, A);
    Wait(t_g);
```

delays. Furthermore an arbitrary delay between 2500-3000 ms is added to simulate the receival of 1 million SHA3-256 hashes of 32 Byte, based on the average network speed of around 100 mbps (Ookla, 2021). The revocations were released on $t = 0$ by a single client. We opted to simulate revocation data in order to allow more emulated clients.

B. Simulation Results

The individual traces are visible in Figure 7. As expected, increasing the t_g leads to higher propagation times. Contrary to expectations, Figure 7a and Figure 7b portray a quadratic run time increase with respect to the number of clients. However, as visible this increase is far less prominent with fewer clients, especially portrayed by Figure 7b. This behaviour can be explained by hardware limitations on the workstation limiting the number of messages between clients, as we noted high CPU usage. The high-interval timings (Figure 7c and Figure 7d) portray more the expected logarithmic-natured runtime, as the increased interval imposes less load on the workstation. Furthermore, it can seen that the increase of n_g leads to lower propagation timings, however, this additionally increases the load on the client. To conclude, the simulation showcases great scalability in the revocation mechanism, however, portrays that hardware constrains must be taken into considerations as parameters that impose higher throughput may decrease overall performance due to overhead in system load.

VI. Implementation & Field Trial

Sections IV & V presented a Self-Sovereign Identity framework based on the prior works by Stokkink & Pouwelse (2018); Stokkink et al. (2020) with the novel fully distributed revocation algorithm and offline verification capabilities. Based on this design, two implementations have been made using the IPv8 protocol stack[4]. 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).

Three semantic layers have been implemented on top of the Kotlin implementation of IPv8[5]. Per authors choice two ZKPs claim types have been implemented: firstly, a ZKP proof allowing arbitrary data and the verification of exact values. The implementation is based on the algorithm proposed by Boneh et al. (2005), allowing verifiable computation through 2-DNF formulae over bits. Secondly, the range ZKP proposed by Peng & Bao (2010), allowing encoding of integer values laying in a specific range. The commitment scheme proposed by Boudot (2000) has been implemented in order to realise this range proof, based on the work by Stokkink & Pouwelse (2018). 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. The code for the reference implementation of these semantic layers is available on the IPv8 repository[6].

Secondly, a mobile client has been implemented in the form of an Android application. This client uses the implementation of the three semantic layers and showcases the usability on smart phones. The application supports all discussed communications per the three semantic layers. In addition, clients can create multi-party communication channels in order to force visibility with one another. This is performed through specialised tokens. The application enables offline verification through the presentation of Claims and attestation through QR-codes. As the Claims can comprise any form of data, the client even supports attestations to pictures; opening up the possibility for digitally attested to passport photographs. The application was validated using a minor real-life trial for the ZKP verification of age of majority (see Figure 8). Further trials were cancelled due to the COVID-19 pandemic. The implementation can be found on the Trustchain `superapp` repository[7].

VII. Performance Analysis

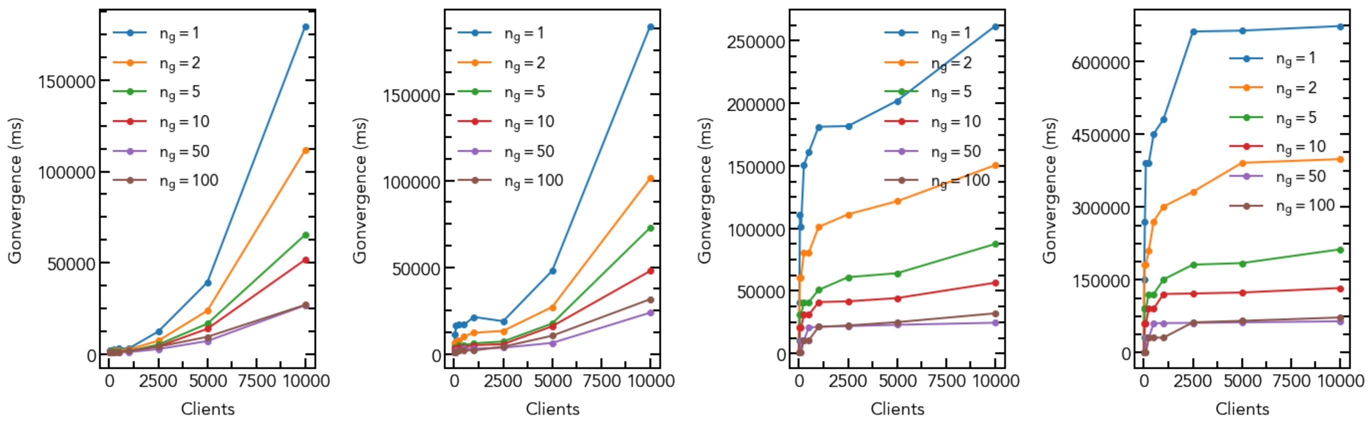
The analysis on smartphones was performed in a test setup measuring the time required to gossip revocations between an Authority and a regular client. For revocations, we generated datasets of 32 bytes SHA3-256 hashes, a format used by the implementation. Revocations were split-up into sets of 1000 in order to minimise the impact of a single packet loss. In order to further prevent packet loss, the gossiping client was restricted to 10 UDP packets per second. For the default parameters, the gossip-interval t_g was set to $100ms$ in order to maximise throughput of gossip. The number of selected peers m_p was set to 5, as IPv8 recommends up to 30 simultaneous connections, such a small amount suffices, especially since the measurements were collected on device basis.

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

[5]For the Kotlin implementation of IPv8, see: `https://github.com/Tribler/kotlin-ipv8`

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

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



(a) $t_g = 100\ ms$ (b) $t_g = 1000\ ms$ (c) $t_g = 10000\ ms$ (d) $t_g = 30000\ ms$

Fig. 7: Simulated propagation times

A. *Revocation Amount*

Figure 9 showcases the revocation scaling in a system of 1 Authority and 3 regular clients ($n = 1, m = 3$). As visible, the propagation time scales linearly with respect to the number of revocations. As visible 1 million revocations take roughly up to 8000 seconds or around 2 hours. As this can be deemed more than two years worth of revocations (HM Passport Office & The Rt Hon Caroline Nokes MP, 2018), we deem this scalability usable.

Compared to the simulation discussed prior, the performance is worse. We note that this can be explained mostly due to communication overhead caused by UDP packet splitting. The tremendous amount of packages led to many packet drops, in turn leading to the loss of specific revocation versions. As the reference implementation naively provides the gossiping client with a lower bound of missing versions, the additional network traffic of already gossiped versions causes more packet losses. This snowballing effect worsens the performance of the algorithm.

VIII. Conclusion

We presented a Self-Sovereign Identity framework which can facilitate the digital identity needs of the European Union.

Fig. 8: Real life trial

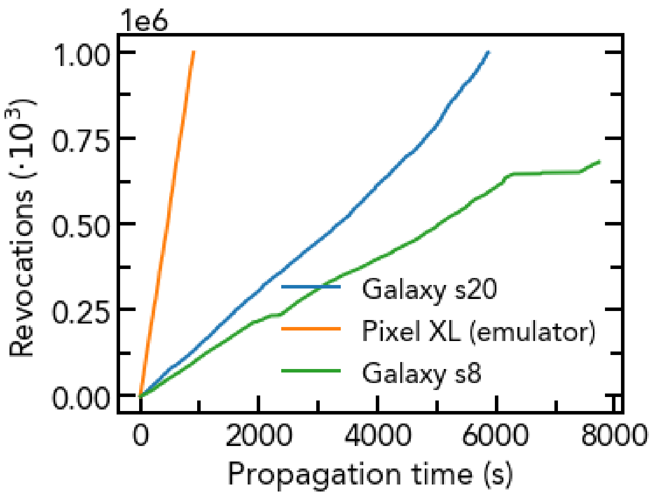


Fig. 9: Propagation timings on smartphones

Most notably containing a first-of-a-kind distributed SSI revocation mechanism, enabling offline verification, capable of fulfilling the missing revocation link in SSI. The model is shown to provide fully distributed reliable revocation through unreliable communication links and showcases usability on smartphones. Privacy is aided through the usage of zero-knowledge proofs and communication with selected peers. A reference implementation for the semantic layer has been created, as well as a mobile client showcasing full feasibility on smartphones. Our small scale trial shows that fully distributed SSI is feasible on modern handheld devices and that this is a promising direction to further explore.

[TODO: Fix references]

References

Abraham, A., More, S., Rabensteiner, C., & Horandner, F. (2020, 12). Revocable and offline-verifiable self-sovereign identities. *Proceedings - 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2020*, 1020–1027. doi: 10.1109/TRUSTCOM50675.2020.00136

Arias, D. (2021, Feb). *Adding salt to hashing: A better way to store passwords.* Auth0. Retrieved from https://auth0.com/blog/adding-salt-to-hashing-a-better-way-to-store-passwords/

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*.

by Design Foundation, P. (n.d.). *Privacy by design foundation.* Author. Retrieved from https://privacybydesign.foundation/irma-explanation/

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

Demers, A., Greene, D., Hauser, C., Irish, W., Larson, J., Shenker, S., … Terry, D. (1987). Epidemic algorithms for replicated database maintenance. In *Proceedings of the sixth annual acm symposium on principles of distributed computing* (pp. 1–12).

Eschenauer, L., & Gligor, V. D. (2002). A Key-Management Scheme for Distributed Sensor Networks *.

European Commission. (2021, Jun). *Eu digital covid certificate.* European Commission. Retrieved from https://ec.europa.eu/info/live-work-travel-eu/coronavirus-response/safe-covid-19-vaccines-europeans/eu-digital-covid-certificate_en

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

Hardman, D. (2018, 2). *HIPE 0011-cred-revocation.* Retrieved from https://github.com/hyperledger/indy-hipe/blob/4fd9db58/text/0011-cred-revocation/README.md

Hardman, D. (2019). What if I lose my phone. Retrieved from https://sovrin.org/wp-content/uploads/2019/03/What-if-someone-steals-my-phone-110319.pdf

HM Passport Office, H., Border Force, & The Rt Hon Caroline Nokes MP. (2018, Jun). *Report your lost or stolen passport.* GOV.UK. Retrieved from https://www.gov.uk/government/news/report-your-lost-or-stolen-passport

IETF. (n.d.). *Public-key infrastructure (x.509) (pkix).* Retrieved from https://datatracker.ietf.org/wg/pkix/

IRMA. (n.d.). *Revocation.* Retrieved from https://irma.app/docs/revocation/

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. Retrieved from https://www.ingwb.com/media/2122048/zero-knowledge-range-proof-whitepaper.pdf

Kwiatkowska, M., Norman, G., & Parker, D. (2008). *Analysis of a Gossip Protocol in PRISM* (Tech. Rep.). Retrieved from http://www.prismmodelchecker.org/casestudies/gossip.php

Laberteaux, K. P., Haas, J. J., & Hu, Y.-C. (2008). Security certificate revocation list distribution for vanet. In *Proceedings of the fifth acm international workshop on vehicular inter-networking* (pp. 88–89).

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*.

Mühle, A., Grüner, A., Gayvoronskaya, T., & Meinel, C. (2018, 11). *A survey on essential components of a self-sovereign identity* (Vol. 30). Elsevier Ireland Ltd. doi: 10.1016/j.cosrev.2018.10.002

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

Ookla. (2021, Jul). *Speedtest global index.* Author. Retrieved 2021-07-21, from https://www.speedtest.net/global-index

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., Jungels, D., Papadimitratos, P., Aad, I., & Hubaux, J.-P. (2006). *Certificate Revocation in Vehicular Networks* (Tech. Rep.). Retrieved from https://www

.researchgate.net/publication/37433732

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

Siftery. (2017, Jan). *Top social login tools compared.* Medium. Retrieved from https://medium.com/@siftery/top-social-login-tools-compared-b350eae26118

Sovrin. (n.d.). *Stewards.* Retrieved from https://sovrin.org/stewards/

Sporny, M., Longley, D., & Chadwick, D. (2019, Nov). *Verifiable credentials data model 1.0.* W3C. Retrieved from https://www.w3.org/TR/vc-data-model/

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).

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

uPort. (n.d.). *uPort Developer Portal.* Retrieved from https://developer.uport.me/

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

Xu, J., Xue, K., Tian, H., Hong, J., Wei, D. S., & Hong, P. (2020). An identity management and authentication scheme based on redactable blockchain for mobile networks. *IEEE Transactions on Vehicular Technology*, *69*(6), 6688–6698.

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