Eemsgolaan 3
9727 DW Groningen
P.O. Box 1416
9701 BK Groningen
The Netherlands

www.tno.nl

T +31 88 866 70 00

TNO report

| Final report

Self-Sovereign Identity: A Comparison of IRMA and Sovrin

Date July 14th 2019

Author(s) Jelle C. Nauta, Rieks Joosten

Number of pages 21 (incl. appendices)
Number of appendices ---
Report number TNO 2019 R11011
Project name Techruption
Project number 060.37572

This research was performed in the context of the self-sovereign identity track of the joint Techruption R&D programme [https://www.techruption.org]. It has been partly funded from PPS-budget for research and innovation of the Dutch Ministry of Economic Affairs and Climate.

Summary

Over a decade ago, Kim Cameron and others dreamed of what was called an Internet Identity Layer [1]; it would do for (the exchange of) (identity) data what IPv4 had done for network transport: make sure that all local solutions could live together to form a globally connected infrastructure. could be exchanged throughout the world in the same way.

Much effort has been put in by many to realize this vision, which is currently referred to by the phrase Self-Sovereign Identity (SSI), and which roughly refers to a set of principles that are generally considered the basis on which a next generation of internet applications will exchange (issue, request, and obtain) personal data [2].

Today, we see tens if not hundreds of initiatives that work with these principles. However, it is still quite difficult to satisfy all of them: [3] surveyed some 50 of them, and identified the three that came closest: uPort, IRMA and Sovrin.

In this document we try to compare two of them: IRMA and Sovrin, the purpose of which is to give our audience an idea about what they are about, what the differences are, and where we stand today. The comparison is rather high-level, and touch upon functionality, protocols and maturity.

We conclude that this area is still evolving (and quite rapidly). Quite a few issues remain to be resolved, most of which perhaps are not of a technical nature, but have to do with usability/user friendliness, legal stuff, adoption and the ability and willingness of businesses to transition to this new way of working.

Contents

1 Introduction

On the internet, nobody knows you're not a dog – unless you can prove it. Many applications, ranging from health insurance claims to mortgage applications, require the exchange of information between a service provider and a user. Such information is needed so that parties can agree what they get from each other, and that each of them can assess whether or not it is a good idea to provide/obtain the service. In order for a party to make a valid decision, it must have valid data, i.e. data that is (sufficiently) true and meaningful.

Especially for high-value transactions, it is generally not a good idea to trust the data that the other party provides at face value, as the other party may be 'a dog' (rogue). The Self-Sovereign Identity (SSI) paradigm aims to contribute here by providing means for each party:

- to request the other party to provide data that is attested to by a third party that the author of that request trusts.
- to respond to such requests by providing attestations that satisfy such request, and
- to verify responses, i.e. to (cryptographically) check that the attestations have been issued by the designated third-parties, and that they have not been altered since the time they were issued – in other words: that it can trust the validity of that data for making decisions with.

SSI employs privacy preserving technologies, which means e.g. that it is hard for parties to link data from different transactions, or for different parties to link information they obtained from users. This makes SSI technologies applicable for situations where information not only must be trusted, but also must be kept private, such as in healthcare and finance.

Current ambitions of SSI, i.e. to support digital transactions that are high-valued, privacy sensitive, user-friendly and legally enforceable, already has quite some history [2]. The ambitions make a comparison of technologies quite difficult, because a thorough comparison, in particular when considering whether or not it is 'fit-for-purpose', would need to take such non-technical issues into account. This report only scratches the surface of the status of two such systems as they stand today.

Some time ago, other comparisons have been made. For example, [3] did a comparison of nearly 50 projects, listing IRMA, Sovrin and uPort as the three solutions that came closest to satisfying the SSI principles. As uPort is converging to Sovrin, we limited ourselves to the first two in our comparison.

This review aims to provide readers with an idea about what different SSI systems may be about. We chose IRMA and Sovrin not only because they are two prominent representatives of SSI systems, but also because one (IRMA) is not based on blockchain technology, and the other (Sovrin) is.

Our comparison is structured into three topics:

1. **Functionality**: What can IRMA and Sovrin do, and what are (currently) their functional limitations?
2. **Protocols**: How do they achieve their functionality? This provides a bit deeper insight into the technologies, which is necessary to understand certain advantages and disadvantages.
3. **Maturity**: what is the current state of the two projects, and how can they expected to develop in the future?

This review is based on a study of available literature, as well as on feedback from experts in the field (see section 7), which includes people that are actively involved in the development of IRMA or Sovrin.

2 Functionality

SSI allows for organizations (credential issuers) to issue credentials to users (credential holders). A (simple) credential is a set of statements (attributes) that the issuer considers to be true of the holder. A holder can use her credential to prove to a relying party (service provider, credential verifier) that the credential was indeed issued to her by the issuer, and also that its contents have not been changed since it was issued. This implies that the relying party can verify these proofs. SSI also has a notion of revocation, i.e. the ability of relying party to verify that a credential has (not) been revoked.

This chapter discusses the differences between Sovrin and IRMA concerning their functionality for the issuer, holder and relying party.

2.1 Privacy

Both Sovrin and IRMA have a strong focus on privacy, with very similar features which we briefly discuss below. Note that our aim here is not to give a comprehensive privacy analysis which would require a much more detailed study of the protocols.

Unlinkability

Neither issuers nor relying parties (even if they collude) can detect repeated use of a holder's credential. Issuers do not even know when or where the credentials that they have issued are being disclosed by their holders. In the case of Sovrin there are also nodes in the network that allow interacting with the ledger. These nodes are queried to verify that credentials have not been revoked, but these queries are not user-related and should pose no privacy risk.

Interestingly, the cryptographic foundations of Sovrin and IRMA are very similar. Both use zero-knowledge proofs with Camenisch-Lysyanskaya signatures [6], although with different cryptographic primitives as the basis. IRMA uses Idemix and Sovrin using the Charm library and elliptic curve signatures [9].

Selective disclosure

Both IRMA and Sovrin allow for holders to disclose individual attributes (key-value pairs) from their credentials (sets of attributes coming from one issuer), and combine these in a single proof for disclosure to a relying party.

Cloud storage

- IRMA stores all attributes in the app. Only the schema is stored online, and (for extra security and revocation) part of the user's private key.
- Sovrin stores the schemas and credential definitions on a public blockchain. Like in IRMA, credentials are kept in an "agent" (software of the holder), which may or may not be in the cloud – but pervasive use of cloud agents is expected in the future. Nevertheless, this remains optional.

2.2 Schemas

Sovrin uses DIDs [4] as identifiers for identity holders. Every DID-owner can create schemas on a public blockchain, and credentials hold references to these schemas. This gives issuers complete freedom in what they choose to publish, but could make it difficult to reap the benefits of using schemas (semantic interoperability). If many different schemas are created that describe essentially the same data, correspondences between such schemas would have to be created. Sovrin's hope and expectation in this regard is that "there will be a natural and organic convergence around the most useful schemas." [4] It seems plausible that legal recognition of certain credentials will promote this.

IRMA's approach to schemas is currently more centralized; the Privacy by Design Foundation has published the schemas on github for various issuers [7], and these schemas are used by the IRMA app using an auto-update mechanism. New issuers must be added by the foundation.

There is some research into decentralizing IRMA using Ethereum [8]. This points in a similar direction to Sovrin's DPKI-approach, but also notes that "there is no immediate advantage of decentralizing the scheme for schemes that have a relatively 'official' (central) status, which is true for the schemes that are most important in the short term (such as Government or organizational schemes)".

2.3 Revocation

The default way for credentials to become invalid is because they expire. However, there are situations in which credentials need to become invalid before their expiration time has elapsed. The process of invalidating credentials before they expire, is called revocation.

IRMA currently only allows users to revoke their credentials, e.g. in case of theft or loss of their wallet. IRMA currently does not allow issuers to revoke credentials they issued. Revocation is done through use of a keyshare server, as explained in the next chapter. A disadvantage of this is that the keyshare server is a single point of failure for use of a credential, potentially preventing all users from accessing their credentials – which could eventually happen through purposeful censorship, hacking or some (natural) calamity. Administering ones keyshare is done through MyIRMA which can be logged onto either by scanning a QR-code with the IRMA app or through a URL sent to the email address that was (optionally) used to register with IRMA.

An issuer that wants to be certain that the IRMA credentials it issues are valid for say 99.9% of the time, will need to make them ephemeral (short-lived), by setting the expiration period to a low value (e.g. 2 weeks), and relying on an automatic, online renewal process that makes this not a user issue. The choice of expiration dates is constrained to fixed points in time to avoid user identification based on expiration dates.

Sovrin on the other hand only allows issuers to revoke credentials. The holder can effectively render the credential obsolete by deleting it from his/her wallet, however. The holder can also ask the issuer to revoke the credential.

The ability to revoke a credential is based on the fact that when it is issued, a revocation ID is both inserted into the credential and added to a so called cryptographic accumulator that is maintained on the Sovrin blockchain [10]. When presenting the credential to a relying party, the user includes a proof that the revocation ID (which is not revealed to the relying party) is still contained in the accumulator. The issuer can now revoke the credential by removing the revocation ID from the accumulator, from which moment on the user can no longer present a valid proof[1]. This method is used to preserve privacy for the holder by preventing correlation by the issuer, the relying party and even the Sovrin nodes.

2.4 Other

IRMA features **attribute-based signatures**, a powerful concept which allows the client to sign (the hash of) any message, effectively attaching a subset of its attributes to it. An example of this is the client signing a contract, attaching her age-attribute to it to prove being old enough to sign a contract. No analogous mechanism is known for Sovrin, but given the similarities between the two technologies, it would probably be possible for Sovrin to implement it as well.

[1] Note that there is a timing issue here: the proof of non-revocation applies for the time at which the user presents the credential to the relying party, while the latter needs such proof to apply for the time it actually uses the information from the credential. As the time between these events increases, so does the risk (of the relying party) of using data that has been revoked.

3 Protocols

3.1 Sovrin

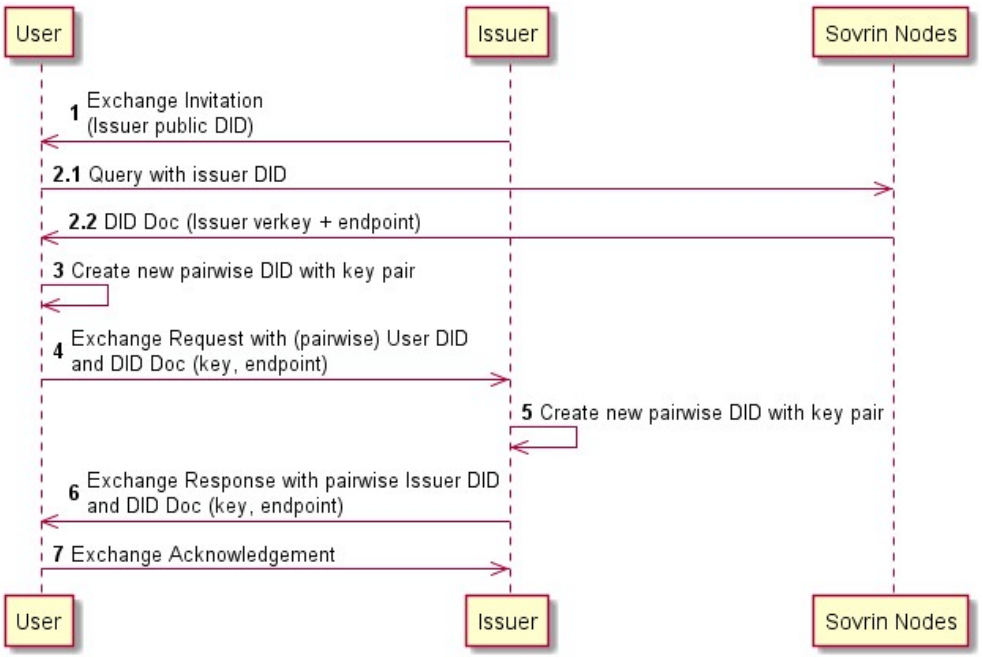
As stated by Sovrin, “The Sovrin Protocol is a set of standards, patterns, and tools that facilitate trusted interactions.”. This protocol has a set of sub-protocols (named **Themis**) for setting up issuers, making connections, and the issuing, proving and revocation of credentials, which relies on other features of the Sovrin protocol such as keys and agents [12].

Some transactions require changes on the Sovrin ledger, in which case they are sent to validator nodes who add them through the Plenum consensus protocol [13]. The following description of Sovrin processes is mainly based on [15, 16].

All parties need to have trusted software (agents) to safely store their cryptographic (Ed25519) keys and interact with each other. Agents use their keys to establish secure communications channels between them, and establish trust using verifiable credentials (bootstrapping trust in issuers might be done using traditional PKI).

Establishing a relationship

The available documentation describes multiple variants of establishing a relationship, corresponding to different use cases and/or evolving ideas about how Sovrin should be used. Originally [14] users had a public pairwise DID (verinym) for each issuer which was written to the ledger. This is no longer the case, Users and Issuers now have off-chain pairwise DIDs (pseudonyms) [15]. The process below is based on [16] and somewhat simplified, assuming an Issuer with a public DID and a User who has only a single agent (wallet). Note that this protocol can also be used for establishing relations between users and relying parties.



1. The Issuer sends the User an Exchange Invitation. This is a plaintext out-of-band message (e.g. using a QR-code).
2. The User queries the Sovrin network for the Issuer's public DID Doc, to obtain its endpoint, verification key (verkey) and authentication information.
3. The User creates a DID and keypair to be used exclusively in interactions with this Issuer.
4. The User sends an Exchange Request to the Issuer, providing her own new DID and DID Doc containing the verkey and optionally an endpoint associated with the DID. The message is encrypted with the Issuer's public verkey.
5. Like step 3, but for the Issuer. The Issuer creates a new DID and keypair to be used only in interactions with the User.
6. Like step 4, but from Issuer to User and including an endpoint for subsequent communication from the User to the Issuer. The message is encrypted with the verkey that the User sent in step 4.
7. User and Issuer now have DID's to identify each other, cryptographic keys for secure communications, and the User has an endpoint to contact the Issuer at. The final acknowledgement message is considered optional but formally completes the sequence for both parties.

This protocol thus establishes a pair of DIDs (and associated keys) that can be held private between the user and issuer, and that they can use to set up 2-sided authenticated (trusted) communications channels for as long as both choose to remember/store the associated data. Using this protocol may make the use of usernames and passwords pretty much obsolete.

Credential issuance

Credential issuance requires the Issuer to first add a credential definition to the ledger [12] by submitting the appropriate transaction [19] to a validator node. Issuance is very straightforward and follows the "Negotiation pattern" [17], resembling the basic business transaction pattern from DEMO [18]. The details in the description below are based on [15].

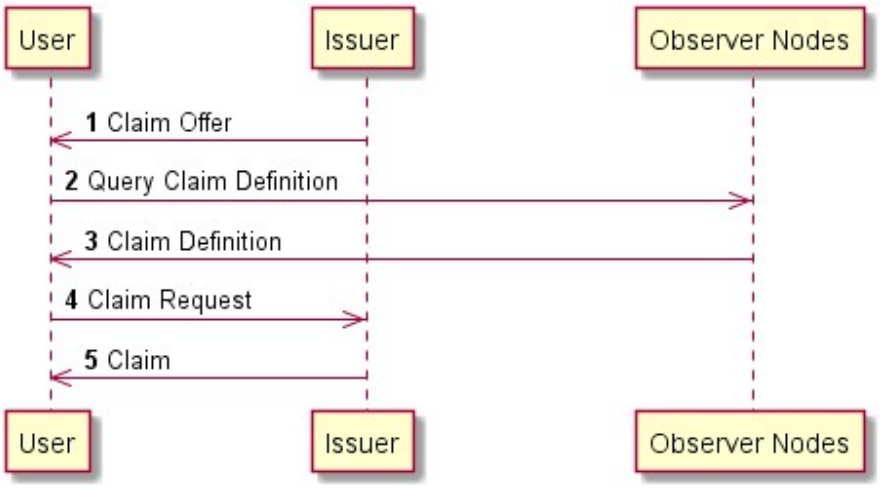


Figure 1: Credential issuance with Sovrin

1. The Issuer sends a “Claim Offer”, referencing the claim definition on the Sovrin ledger to the User. If the User provided an endpoint to the Issuer, this can be done actively, otherwise the User must poll the Issuer for it.
2. To be certain what kind of claim the User is about to be issued, she may query the Observer Nodes to get the definition from the ledger.
3. The Nodes return the credential definition.
4. The User sends back a “Credential Request” referencing the Claim Offer, including a blinded link secret and blinded proving secret.
5. The Issuer sends the Claim (containing the blinded secrets from the User, a number of attributes and a reference to the on-ledger Credential Definition) to the User. The Claim is signed using the Issuer’s public DID.

Relying parties that require the user to present (attributes from) multiple credentials that were issued by different issuers, will need proof that these credentials have actually been issued to one and the same user. Users can provide such proof by having issuers include a blinded version of their private link secret in a credential that they issue; they can then construct a (zero-knowledge) proof for any combination of credentials, that proves they all contain a blinded version of the same link secret (without revealing the link secret itself).

Credential usage

Usage is pretty straightforward, as shown in Figure 2. closely resembles issuance, it simply lets the User ‘issue’ a proof to the relying party. The first step (offering to send a proof) is omitted, i.e. the Relying Party initiates the interaction by sending a request for the proof.

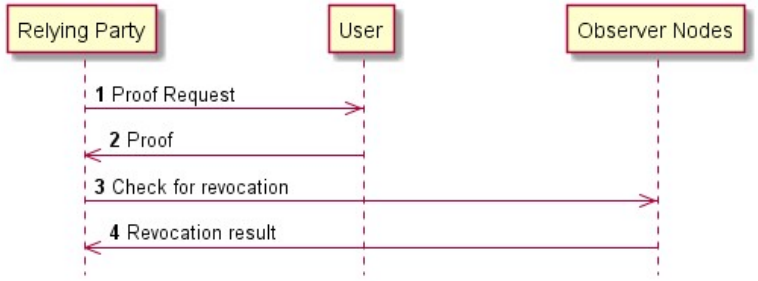


Figure 2: Credential usage with Sovrin

1. The relying party sends a “Proof Request”, asking the user to provide cryptographic evidence for some attributes (e.g. that she is over 18) or for actual data (e.g. her legal name). Proofs and data may be specified to originate from different credentials and different issuers.
2. The user response with a proof (also called “presentation” or “presentation token”) that contains the requested proofs and data, as well as additional proofs, i.e. that the credentials that the proofs/data originated from are all associated with the same link secret, that none of the credentials have been revoked at that point in time, and that the agent producing the response may do all this on the User’s behalf.
3. The relying party checks the revocation proof against the accumulator on the ledger, and
4. obtains the result (i.e. whether or not the revocation proof that the user provided is valid).

Revocation

As mentioned before, Sovrin implements revocation by issuers using cryptographic accumulators, which are mathematical constructs to enable issuers to revoke credentials [10]. For this the Issuer assigns serial number to the credentials that it issues. These serial numbers may be added or removed from the revocation registry on the blockchain. Whenever a verifier wants to check that the credential from which a proof was constructed has not been revoked, she queries the blockchain. The use of cryptographic accumulators ensures that the proofs cannot be linked to the serial numbers in the revocation registry.

Recovery
To prevent problems when a User loses the device on which her wallet is stored (typically a phone), Sovrin offers a few options [15]:

- Using a secondary agent, e.g. installing the Sovrin software on a laptop and adding its keys to her agent policy registry.
- Creating a "paper agent" by printing out keys and DIDs, analogous to a paper wallet for crypto currencies.
- Using multiple agents and requiring some proportion of them to update her policy registry. This prevents a single agent from having too much power, in case it is compromised.
- Using sharding, the User can distribute a key over a group of trusted individuals who may cooperate to reconstitute the key, in case the User loses her normal key.

3.2 IRMA

The IRMA protocol is described in [20, 21] and summarized below using the figures from the description page. In an IRMA-session, there are four components:

1. **Service Provider**: the component that is asked to provided either an issuer of credentials or someone who needs attributes from the client.
2. **ServiceProviderIRMAServer aka the API-server**: the central component in the system, handling all interactions with the client. The API-server must be trusted (possibly owned) by the Service Provider. API-server responses to the Service Provider are trusted via RSA-signatures.
3. **EndUserToken aka the Client**:
4. The **end user**, who only interacts with the client to approve transactions.

The API-server and the Client are the most important components, the Service Provider only refers the latter to the former and then waits for the result.

Credential Issuance

The interactions during issuance are shown in the figure below. The ServiceProvider is the issuer of the credential in this case.

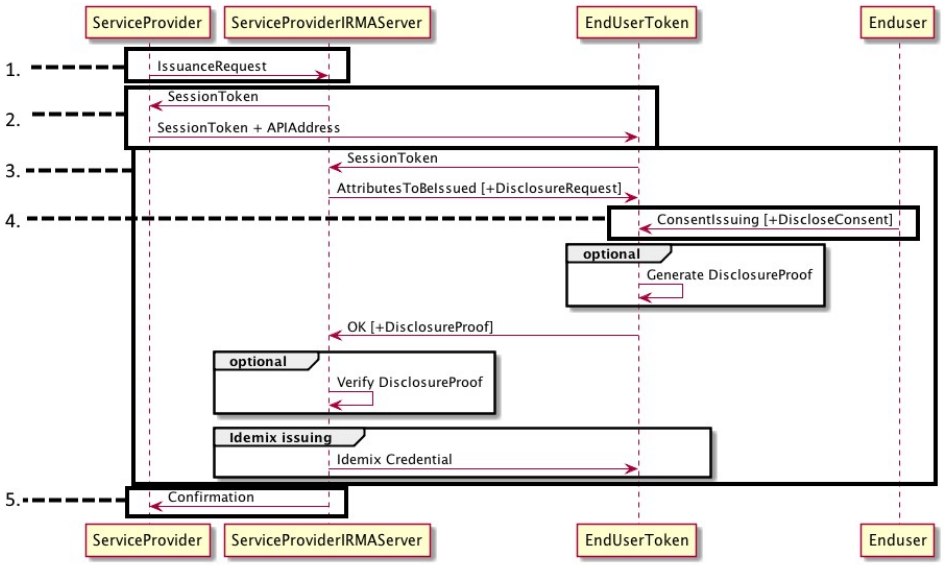


Figure 3: Credential issuance with IRMA

1. IssuanceRequest specifies which attributes are needed, and which attributes will be issued.
2. SessionToken is obtained from the API-server by the Service Provider to refer the Client to the API-server, identifying the Service Provider's IssuanceRequest for this Client. This is all the Service Provider does, it just waits for the confirmation of issuance after this.
3. The Client contacts the API-server. Based on the IssuanceRequest that the API-server received, the Client may disclose attributes and receive new credentials.
4. The only role of the end user is to approve the transaction, i.e. disclosing attributes and receiving the new credential. This is typically just pushing a button in the IRMA app.
5. Once the transaction is complete, the API-server returns a confirmation response to the Service Provider to conclude the process.

Credential usage

Usage (attribute verification) is very similar to issuance as explained above, in terms of the protocol. There are a few differences:

- The ServiceProvider is not an Issuer but someone who needs attributes from the Client.
- Instead of an IssuanceRequest, a DisclosureProofRequest is sent to initiate the process.
- The optional step regarding disclosure from the issuance process is performed (during Usage, credentials are disclosed by definition so this requires User consent).
- No "Idemix issuing" takes place.
- In the last step, rather than a simple confirmation the API-server sends the actual attributes back to the Service Provider.

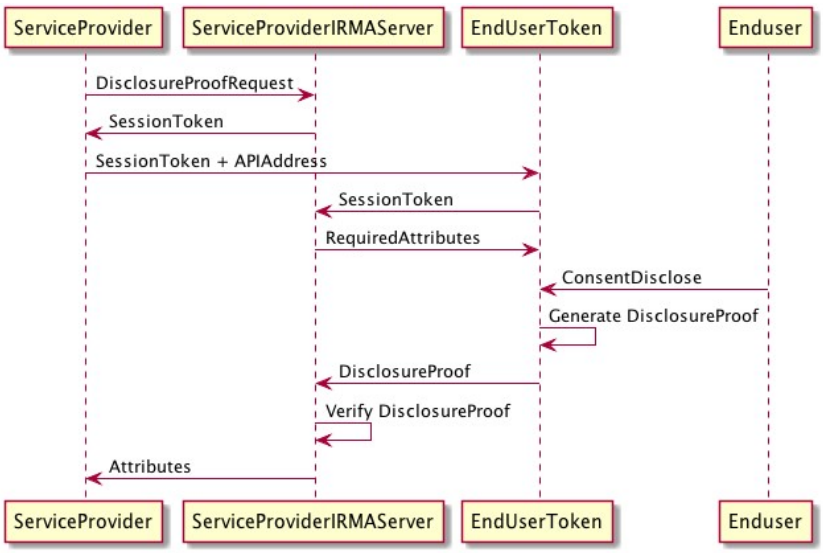


Figure 4: Credential Usage with IRMA

Revocation

In the previous section, the generation of a disclosure proof was simplified to occur only in the Client. In reality, the Client makes use of a keyshare server [22], which holds part of the credential holder's private key and is contacted after the End User gives disclosure consent. The holder must enter a PIN-code upon starting the IRMA-app to unlock the keyshare. A good description of how the disclosure proof is created and can combine attributes from different credentials is given in [23].

Revocation is done via a web interface at https://privacybydesign.foundation/myirma/ (for the keyshare server of the foundation). The User logs in here with their MyIRMA-account, which is created upon installing the wallet app. Note that revocation is an all-or-nothing mechanism, no individual credentials can be revoked by the User (which makes sense, since either none or all of their credentials are compromised).

Recovery

There has been some recent work with a proof-of-concept in IRMA to enable recovery and transferring of wallets to a new device [24]. This is done by creating a backup file of the User's wallet and using the keyshare server mentioned above to decrypt it and revoke the wallet on the old device.

4 Maturity

4.1 Practical use

4.1.1 IRMA

In terms of technological maturity, IRMA is slightly more convincing, having a working wallet app for several years now and GDPR-compliant government-issued credentials by the municipality of Nijmegen [25] in the Netherlands since last year. It also has features such as attribute-based signing and credential-only proofs (disclosing possession of a credential without disclosing any of its attributes). Transfer of a wallet to a new device has recently been demonstrated but is not production-ready yet [24].

For service providers, adopting IRMA is relatively straightforward, though it does require hosting an IRMA server, adapting one's login page/portal. And of course, users must install the IRMA app. The fact that all of this can be experimented with easily (also using the demo IRMA server) without depending on external parties makes IRMA quite attractive to get started with.

For credential issuers, it is necessary to contact the privacy by design foundation, which oversees the addition of new issuers. Candidate credential issuers undergo a vetting process and, if successful, have their public keys added to the scheme on github [7] which is used by the IRMA app and IRMA server. Apart from that, roughly the same steps as for service providers are required.

4.1.2 Sovrin

Sovrin is a much newer project, and technologically more complex due to its use of blockchains. Wallets are currently being developed [27], most notably the Connect.Me app [28]. Development of the app was initially open source [36] but the recent "General Availability" release [26] is a closed source continuation of this. The app is available from the Google Play store and iTunes App store, and an online tutorial guides the user through its functionality. Other wallets in an earlier stage are Mattr [29] and Streetcred [30].

Sovrin is used in a sizeable pilot project called Orgbook in Canada by the governments of British Columbia and Ontario [31]. The project lets companies and other organizations easily register to obtain credentials and be publicly searchable, reducing their previous bureaucratic burden. At the time of writing, British Columbia reports that there are almost 2 million held credentials in their project [32]. Future plans include the issuing of permits and licenses through the Orgbook.

At present, adopting Sovrin can be a bit challenging for both service providers and credential issuers. Since Sovrin and Hyperledger Indy are such complex projects (and very much a work in progress), documentation is scattered and often outdated. Interacting with the Sovrin (identity) ledger is doable using indy-cli but requires some technical know-how. Experimentation requires either a lot of terminal commands and copying and pasting of output or paying a service provider to help with more structured proprietary demos.

An advantage for service providers that Sovrin has compared to IRMA is that they do not need to host any servers (or at least simpler ones in the form of Sovrin agents [33]). A recent redesign of the IRMA server has made it easier to setup, but this still involves cloning multiple github repositories, non-trivial install instructions and configuration of a database. Sovrin agents simply query the Sovrin ledger whenever they wish to verify someone's credentials. IRMA's approach to encourage

Before an organization can act as a credential issuer in Sovrin, it must obtain the role of Transaction Endorser, which it can be granted by Sovrin Foundation. There is also a test network for which everyone gets accepted as a trust anchor, without the vetting process that is required for the main network. The purpose of this is not to allow only trustworthy issuers but to comply with global regulations such as GDPR [34]. Sovrin aims to allow anyone to become an issuer, but it is up to the verifier to decide whom to trust. The Sovrin Network is also considering introducing a cryptographic token which would allow open public access, meaning anyone could become an issuer without the intermediary of a Foundation or even a steward to give permission.

4.2 Project organization and outreach

In terms of international presence and project organization, Sovrin does very well, with Stewards on every continent to run the network and partnerships such as with the Finnish TrustNet consortium [37], pilot projects in Canada and winning the "Greatest Social Impact" Know 2019 award [35]. IRMA is well-known in the Netherlands, recently winning two awards [38, 39], but has little activity abroad. A live overview of its registrations and issued credentials is available online [40].

4.3 Funding

IRMA is largely built by volunteers and funded by subsidies and donations (most notably by the company Alliander) [41], though it may gain more revenue in the future by charging issuers for maintaining their schemas and/or issuing their credentials. IRMA's yearly expenses in 2017 were in the order of €30.000. At the time of this writing no report for 2018 was available yet, but the foundation reported a budget of €300.000 for 2018 upon inquiry. To ensure long-term stable and secure hosting of its keyshare server, the Privacy by Design Foundation has a partnership with SIDN [42].

The main source of the Sovrin Foundation's income is corporate contributions. These contributions come from two sources; the Sovrin Alliance [43], which is funded by IBM and others in the form of both code and financial contributions; and Evernym, which provided $3 million of seed funding. Sovrin's funding model is based in part on the Sovrin token [44], a cryptocurrency that is envisioned to eventually incentivize actors in the network. There is little public information available concerning the creation and distribution of this token and Sovrin would not comment on this either due to regulatory considerations, but it stands to reason that the success of Sovrin depend to some degree on the future stability of this token. Sovrin's yearly expenses in 2018 were around $2.5 million (over €2 million) [45], more than 60 times as large as IRMA's, which reflect the global scope of the Sovrin project.

5 Compatibility

We will briefly discuss the possibility of using both IRMA and Sovrin (and possibly more systems) together. For this to be user-friendly, a single app would have to work transparently with both Sovrin and IRMA credentials (and possibly others). A straightforward way for service providers would be to accept multiple types of credentials, or alternatively issuers would have to issue the same credential for multiple systems.

As a bad alternative to using the systems in parallel, one could imagine both systems creating credentials based on those of the other. For example, within Sovrin there could be an "IRMA trust anchor" (controlled by either IRMA or Sovrin). This trust anchor would create a schema and credential definitions within Sovrin that would mirror those within IRMA. It would then issue Sovrin-credentials after obtaining a corresponding IRMA proof from the user. These Sovrin credentials would contain a reference to the original issuer in the context of IRMA, requiring a verifier to resolve this reference and to somehow decide whether it trusts this issuer.

The philosophical problem with this is that it would centralize all the IRMA credentials, making the IRMA-issuer the issuer of all credentials, rather than the organizations that issued them originally.

There would also be technical problems arising from synchronization of the two systems (both in issuance and revocation) and resolving references to the original IRMA-issuers. The latter would require a way of mapping the public key that identifies an issuer in the IRMA domain to the DID that identifies the issuer in the Sovrin domain. The same such mapping/reference resolution would be required for credential definitions and schema's, with the added requirement of semantic operability; verifiers have to automatically interpret credentials from both systems and be able to decide whether they are equivalent (in this example: given that I accept this IRMA credential for a particular purpose, should I also accept this Sovrin credential?)

Finally, there might be a problem of incentives. Since the success of an identity system benefits from the utility it gives to its participants, no organization would put effort into porting its identities to an ultimately competing system.

Although the considerations above are just a quick thought experiment, they do illustrate the challenges that would arise from making multiple identity systems compatible. It therefore seems unlikely that multiple systems will gain and keep widespread adoption, unless they can offer different benefits to their participants. The use of open standards greatly helps in reducing the dependence of participants on any system in particular and enables them to adopt other systems more easily. Sovrin's use of the emerging W3C's DID and verifiable credentials standards [46] is quite promising in this respect. IRMA is fully open source and thus also allows for easy interoperability, but use of widespread standards would be beneficial for easier interoperability with other self-sovereign identity systems in the future.

6 Conclusions

IRMA and Sovrin are similar in many respects. The main difference from a technological standpoint lies in the treatment of public identities (issuers), schemas and revocation. Sovrin uses a public, permissioned blockchain with impartial and geographically diverse validator nodes. IRMA is currently relying on PKI, in which the foundation decides who may issue credentials, but it may also transition towards a more distributed trust model in the future.

An advantage that Sovrin offers over IRMA is the revocation of credentials by the issuer. Direct revocation by the user is only possible in IRMA, but comes at the cost of dependence on a central keyshare server.

In choosing which technology to use, it seems that factors other than functionality could be more important, such as ease of use and technological maturity (in which IRMA currently comes out ahead due to its longer history and lower technical complexity), expected adoption (where Sovrin seems more promising due to its international outlook and sheer size), and stability in the future (difficult to predict). The legal validity of credentials is also an important aspect for many applications which differs between countries and may take a lot of effort to achieve. In this regard Sovrin has come a long way in Canada while IRMA currently has more traction in the Netherlands.

The long-term use of multiple identity systems in parallel requires the use of open standards for identifiers and credentials to allow for interoperability between these systems and reducing lock-in of users to any one system. Both IRMA and Sovrin are open in this respect, but Sovrin's use of W3C standards currently emphasizes this more.

7 Acknowledgements

We would like to thank Oskar van Deventer, Riley Hughes, Bart Jacobs and Sietse Ringers for their insightful reviews and fruitful discussions in the creation of this report.

8 References

1. http://www.identityblog.com/?p=352
2. http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html
3. https://www.slideshare.net/TommyKoens/matching-identity-management-solutions-to-selfsovereign-identity-principles
4. https://w3c-ccg.github.io/did-spec/
5. https://github.com/sovrin-foundation/protocol
6. https://eprint.iacr.org/2001/019.pdf
7. https://github.com/privacybydesign/pbdf-schememanager
8. https://github.com/timenolthof/irmaethereumscheme
9. https://www.evernym.com/wp-content/uploads/2017/07/Sovrin-Getting-Started-Guide.pdf
10. https://sovrin.org/wp-content/uploads/AnonCred-RWC.pdf
11. https://github.com/sovrin-foundation/protocol/tree/master/janus
12. https://github.com/sovrin-foundation/protocol/tree/master/themis
13. https://sovrin.org/wp-content/uploads/2018/03/The-Technical-Foundations-of-Sovrin.pdf
14. https://github.com/hyperledger/indy-node (commit 071549276174835f24bb5d282b069510ec1e1339)
15. https://sovrin.org/library/how-dids-keys-credentials-and-agents-work-in-sovrin/
16. https://github.com/hyperledger/aries-rfcs/blob/master/features/0023-did-exchange/README.md
17. https://github.com/sovrin-foundation/protocol/blob/master/patterns/protocol-patterns.md
18. https://en.wikipedia.org/wiki/Design_%26_Engineering_Methodology_for_Organizations
19. https://github.com/hyperledger/indy-node/blob/master/docs/transactions.md#domain-ledger
20. https://credentials.github.io/protocols/irma-protocol/
21. https://irma.app/docs
22. https://credentials.github.io/protocols/keyshare-protocol/
23. https://irma.app/docs/keyshare-protocol/
24. https://privacybydesign.foundation/meeting-slides/slides-8-3-19/derksen-8-maart-2019.pdf
25. https://www.nijmegen.nl/nieuws/app-irma/
26. https://www.evernym.com/blog/connect-me-sovrin-digital-wallet/
27. https://dutchblockchaincoalition.org/uploads/pdf/Blockchain-Sovrin-rapport.pdf
28. https://connect.me/
29. https://www.mattr.global/
30. https://app.streetcred.id/
31. https://www.hyperledger.org/blog/2019/03/11/reducing-government-red-tape-british-columbia-creates-new-business-identity-model-with-hyperledger-indy
32. https://orgbook.gov.bc.ca/en/home
33. https://github.com/hyperledger/indy-agent

34. https://blog.tykn.tech/digital-identity-management-in-the-context-of-gdpr-sovrin-43028247378b
35. https://2019.knowidentity.com/2019/03/27/announcing-know-2019-award-winners/
36. https://github.com/sovrin-foundation/connector-app
37. https://blog.sovrin.org/sovrin-foundation-and-finlands-trustnet-join-forces-to-build-a-trust-network-for-distributed-cc3dd767892e
38. https://www.privacy-web.nl/nieuws/brouwer-prijs-naar-privacy-by-design
39. https://www.privacyfirst.eu/actions-1/663-winners-of-the-2018-dutch-privacy-awards-announced.html
40. https://metrics.privacybydesign.foundation/grafana/d/000000011/irma-dashboard?orgId=1
41. https://privacybydesign.foundation/pdf/jaarverslag-2017.pdf
42. https://www.sidn.nl/en/news-and-blogs/why-would-you-share-more-data-than-you-need-to
43. https://sovrin.org/join-the-sovrin-alliance/
44. https://sovrin.org/library/sovrin-protocol-and-token-white-paper/
45. https://sovrin.org/2018_report.pdf
46. https://akasha.org/blog/2019/04/16/akasha-joins-w3c