Sovrin: What Goes on the Ledger?

A white paper from Evernym in cooperation with the Sovrin Foundation.
An overview of what's on the Sovrin distributed ledger and why.

Andrew Tobin, Evernym
Updated September 2018
Originally written April 2017

evernym.com
sovrin.org

Revision History

April 2017	Original version	Andrew Tobin
June 2018	Change “claims” to “credentials”. Introduced private & public DIDs. Updated link to DID spec. Added section on agent authorisation policies	Andrew Tobin
October 2018	Minor changes Replace identity “owner” with identity “holder”	Andrew Tobin

Executive Summary

One of the most common questions about Sovrin is, “What goes on the ledger?” This paper answers that question and explains several key decisions the Sovrin Technical Governance Board (TGB) has made about why data should or should not go on the Sovrin Ledger.

Sovrin’s focus on decentralised self-sovereign identity is built on a foundation of Privacy by Design. Every design decision is intended to increase security and privacy while reducing undesirable impacts and correlation risk.

The Sovrin ledger is designed specifically to be a public utility which can be used by any person or organisation. This has huge benefits in terms of scale, flexibility and adoption—full public accessibility ensures identity can be truly independent and self-sovereign.

As a result, some of these policies differ substantially from other blockchain identity implementations where identity is a “bolt on” rather than the sole purpose of the ledger itself.

Note: Please see the Sovrin Glossary for the definition of any unfamiliar terms in this document.

Background

Distributed ledgers are increasingly being seen as a way to deliver self-sovereign identity—a privacy-protecting digital existence for people, organisations and things that is separate from any data silo. Self-sovereign identity is owned, managed and controlled directly by “identity holders” rather than third parties.

The Sovrin Network, launched by the Sovrin Foundation in September 2016, is the world’s first distributed ledger engineered specifically for self-sovereign identity. Further detailed documentation about Sovrin can be found in the Sovrin library.

Self-sovereign identity is the solution to a fundamental flaw in the design of the Internet: it was built without an identity layer. As a result, every service must build its own data store ("silo") to store and secure customer details. Huge silos have amassed data for billions of users, each one able to monitor, suspend, delete and remove any of its users at will.

This approach has led to security- and privacy-damaging results, as seen by the ever increasing scale of hacking and breaches, tracking, and unauthorized data sharing that is so commonplace in today's Internet.

With the emergence of distributed ledgers, it has become possible to create a public global ecosystem where an individual can own and control their own persistent Internet identity, rather than "rent" tens or hundreds of identities from different silos. With distributed ledger technology, anyone in the world can have a self-sovereign digital existence.

Self-sovereign identity brings benefits that will transform the way online services are developed. As one of the first companies to see this potential, Evernym developed the original code base for what is now Sovrin. Realizing that it had to become a global public utility to deliver the full potential of self-sovereign identity, Evernym donated this code base to the Sovrin Foundation, who in turn contributed it to the Linux Foundation to become the Hyperledger Indy project. The Hyperledger Indy Node project is now the standard open source code base that is run by all stewards of the Sovrin network to maintain the Sovrin public ledger under the governance policies in the Sovrin Trust Framework. The Hyperledger Indy Agent project is the standard open source code base run by developers of the "off-ledger" distributed agents that are the key to interoperable exchange of privacy-preserving digital credentials that can operate at Internet scale.

Identifiers, Public Keys and Endpoints

Decentralised Identifiers

At its heart, the Sovrin Ledger is a registry for **Decentralised Identifiers** (**DIDs**) and their associated public keys and communications endpoints. DIDs are a new type of identifier designed for cryptographically-verifiable digital identity that is "independent" or "self-sovereign", i.e, fully under the control of the identity holder and not dependent on any centralised registry, identity provider, or certificate authority. Developed for distributed ledger applications, DIDs are the foundation for decentralised identity management. The DID specification is published by the W3C Credentials Community Group and has already achieved wide acceptance in the self-sovereign identity ecosystem.

Each DID resolves to a **DID document**, a JSON-LD file that contains all the metadata needed to prove ownership and control of a DID as well as share the cryptographic keys and resource pointers (endpoints) necessary to initiate trusted peer interactions between Sovrin entities. The Sovrin protocol determines how a DID is registered, resolved, updated, and revoked on the Sovrin Network.

Because the DID document contains the current verification key (aka public key) used by the identity holder, and because the Sovrin Ledger provides cryptographic verification of the current DID document, users can rely on Sovrin to return the current verification key for any DID. Doing this without needing to rely on any centralized authority solves one of the biggest issues with existing PKI (public key infrastructure): knowing which key is currently authoritative for a particular identity holder.

Agent Endpoints

A DID document also contains a pointer to a **service endpoint**. The endpoint is the network address the identity holder uses for further communication, such as a URL or an IP address. It enables two identity holders to communicate directly with one another in a private, secure peer-to-peer interaction, DID-to-DID, with no intermediaries.

With Sovrin infrastructure, the service endpoint for a DID will point to an **agent**. An agent is a software program or process acting on behalf of an identity holder to facilitate interactions with other agents or the Sovrin Ledger. With a Sovrin DID, an agent has a persistent address for sending, receiving, storing or routing encrypted messages to its identity holder (much like email or instant messaging, except encrypted and decentralized).

Enhancing Privacy With Pairwise Pseudonymous DIDs

Using a single unique identifier for all transactions has serious privacy implications for an identity holder, since it would enable them to be strongly correlated across all of their transactions. In non-Sovrin distributed ledger applications, where such transactions are written permanently to the ledger, this approach results in an immutable public record of all the interactions of an identity holder.

With Sovrin, the default is for every identity holder to establish a unique DID for every relationship they have. These are called **pairwise pseudonymous** DIDs and they are tremendously powerful. They allow an identity holder to keep their interactions with one party completely separate from any other party. The identity holder is the only person able to map and correlate their pairwise pseudonymous DIDs (each of which also has a pairwise pseudonymous public key and agent endpoint as well). This means the relationship a person establishes with their bank is entirely separate to the one that they establish with a retailer, so neither the bank nor the retailer can use the DIDs to correlate the identity holder's activities with the other.[1]

Public and Private DIDs

Of course, Sovrin identity holders may also choose to have one or more DIDs and associated DID documents added to the Sovrin public ledger. For example, a party such as

[1] This does not by itself mean that the identity holder cannot be correlated, particularly if the identity holder chooses to (or is forced to) share other identity attributes ("claims") which enable correlation. However it does ensure such correlation is not enabled by the Sovrin identity infrastructure itself.

a financial institution who wishes to establish a strongly verifiable public identity can do so by writing a public DID. The associated DID document would typically have multiple service endpoints that, in addition to their agent endpoint, might include the financial institution's public website or other data establishing the institution's name, address, opening hours, and other information that a potential customer can use to establish a relationship.

A DID added directly to the Sovrin public ledger is called a **public DID**, whereas a pairwise pseudonymous DID shared and stored privately "off-ledger" between the agents for two identity holders is called a **private DID**. The ability for Sovrin infrastructure to support both is fundamental to both its Privacy by Design architecture as well as its ability to scale.

Public DIDs are needed first and foremost by issuers of credentials. They are stored on the Sovrin public ledger so that a verifier who receives proof of any credentials is able to look up the public key of the issuer (plus the credential definition and revocation registry—see below) in order to validate the proof.

By contrast, private DIDs are the default in private relationship between two identity holders, where no one else needs to even know about the relationship, let alone the two DIDs being used. There is no need for this pairwise pseudonymous DIDs to be stored on the public Sovrin ledger. If one party changes their public key or agent endpoint, they simply tell the other party the new keys and/or endpoint.[2]

Keeping private DIDs off the Sovrin public ledger has one other massive advantage: scale. Storage of private DIDs and DID documents and communications between pairwise DID connections can take place entirely off ledger, dramatically reducing load and taking full advantage of today's ubiquitous cloud computing infrastructure, while still protecting the privacy and integrity of each identity holder.

As of this writing (September 2018), mechanisms for keeping pairwise private DIDs securely synchronised between two agents are being designed in the open source Hyperledger Indy code. These **microledgers** maintain the state of the relationship between the two parties in order to ensure consistency and prevent attacks.

Example of Public and Private DID Usage

This section explains at a high level how public and private Sovrin DIDs could be used to establish and protect digital relationships.

Independent identity holder Andy wants to use a Sovrin identity to apply for a loan with Seagull Bank. Seagull Bank publishes their public DID on their website and all their literature using a QR code. Andy sees a particularly attractive offer on a poster in a bus station, and scans the QR code with his phone.

[2] In the rare case of two agents "losing touch" because they both moved service endpoints at exactly the same time, the Sovrin public ledger can still provide a privacy-respecting "dead drop" service to enable the agents to re-establish their connection. This service is currently under development in the Hyperledger Indy project.

Andy's Sovrin wallet app recognises the QR code as a Sovrin DID. It accesses Sovrin to look up the DID, and retrieve the current DID document. It verifies that the DID document is valid and signed by the holder of the DID. Within the DID document is the endpoint for the bank's agent.

Andy's Sovrin wallet app then automatically creates a new private DID for Andy's relationship with Seagull Bank. This private DID will have it own private DID document containing a new public verification key, plus the unique agent endpoint that he wants Seagull Bank to use. Andy's Sovrin wallet app then, through Andy's agent, contacts the bank's agent endpoint, peer-to-peer with no intermediaries, and establishes a new connection with the bank using this private DID, DID document, and agent endpoint that are shared only with Seagull Bank.

Seagull bank reciprocates, giving Andy back a private pairwise pseudonymous DID and DID document just for this new private relationship that Andy can verify using the bank's public DID and DID document that are on Sovrin. The bank and Andy now have a secure, private way of communicating whenever they need it. Seagull bank can then use this channel to send Andy specific details on their loan offer. Andy can verify this using the private DID and DID document the bank supplied him with. The bank can do the same thing with Andy's private DID and DID document when he responds to the bank's offer.

Andy and Seagull Bank can use Sovrin first to find one another and then to negotiate a completely private channel that is unique to that particular relationship and has no intermediaries. They can then share verifiable credentials over this secure channel in order to establish the necessary level of trust to do business. This is like Andy calling the bank and the two of them instantly inventing a brand-new language together—that only Andy and the bank know. If someone were to tap the line and listen, they wouldn't understand a thing.

In this example, Sovrin is only used for the lookup, discovery and verification of the bank's public DID. Then each party establishes a private, pairwise pseudonymous DID with the other. From that point on, all further communication and data exchange takes place off-ledger. This ensures both greater privacy and stronger security as neither party is reusing a public key that they have used with any other party. It's also much more scalable because the Sovrin ledger is only involved with the initial establishment of the relationship.

Schemas and Credential Definitions

So far we have talked about the "cryptographic layer" of trust establishing using DIDs and DID documents.

In the example above, it is important to note that when Andy and Seagull Bank established their private connection, it is not the pairwise pseudonymous DIDs that establish overall trust in this new connection. They establish *cryptographic trust*—the assurance that each party has a secure private channel with the other. But by themselves they do not establish the *human trust* that Andy is dealing with Seagull Bank and Seagull Bank is dealing with Andy.

Human trust can be achieved through the exchange of cryptographically- verifiable proofs of digital credentials owned by each party—an exchange that takes place entirely off-ledger, for both privacy and scalability.

To support the interoperable exchange of verifiable credentials, the Sovrin ledger stores two specific objects: **schema definitions** and **credential definitions**.

A schema definition is a machine-readable definition of a set of attribute data types and formats that can be used for the **claims** on a credential. For example, a schema for creating passport credentials would include definition of attributes such as given name, family name, date of birth, passport number, etc. A schema definition can be used by many credential issuers and is a way of achieving standardisation across issuers.

Once a schema definition has been written to the Sovrin ledger, it can now be used by a credential issuer (bank, passport office, university, employer, etc.) to create an issuer-specific credential definition that is also written to the Sovrin ledger. This data structure is an instance of the schema on which it is based, plus the attribute-specific public verification keys that are bound to the private signing keys of the individual issuer. This approach enables an issuer to re-use an existing schema, and enables a verifier who receives a proof containing data from the issuer to look up the issuer's credential definition on Sovrin, obtain their verification key(s) and verify the origin and integrity of that data. Two notes about credential definitions:

1. A single credential definition can incorporate attribute data types from multiple schemas. This encourages the reuse of the same data types across multiple credentials for maximum interoperability.
2. While we normally think of credentials being issued by third-parties such as government agencies, universities, etc., credentials can also be self-asserted. In this case the credential is issued by the identity holder to describe him/her/itself. For example, Andy may issue a credential that he is a fan of Manchester United. He does this entirely on his own authority—Manchester United does not need to be involved.

Enabling credential developers and issuers to publish their own schemas and credential definitions directly to the Sovrin ledger has significant advantages. First of all, because there is no central authority or gatekeeper, anyone can define and issue new types of credentials--credential exchange is completely decentralized in Sovrin. Even the smallest organization, or even individuals, can create new types of credentials that can be instantly issued and verified by anyone in the world. The cumbersome hub-and-spoke model of traditional attribute and data exchange is replaced by a flat layer where credential definitions are driven by utility, agility, and flexibility. New and valuable definitions can be fostered organically to be used both within and across industries.

It will also be possible to build tools to enable credential schemas and definitions to be searchable. A Sovrin user may wish to find out who issues credentials containing certain data types, e.g., finding out which banks issue credentials containing a proof of banking relationship (which can be very useful when renting a house). The person may then elect to

set up a bank account with one of those banks, rather than one that does not issue Sovrin credentials.

Revocation Registries

The more valuable a credential, the higher the chances that the issuer of that credential will need to be able to revoke it if the holder of the credential no longer qualifies for the credential. A common example is a driving license: there are conditions under which it needs be revoked to remove the holder's right to drive.

A blockchain identity system that places credentials (or credential hashes) directly on an immutable blockchain has a hard time meeting this requirement. But the Sovrin Network follows a simple rule: **no credentials are written—even hashed or encrypted—to the Sovrin ledger**. So neither credentials themselves nor hashes of credentials are ever stored on the Sovrin ledger. They are only issued and exchanged off-ledger between Sovrin agents and their respective wallets.

The Problem with "Phone Home" Revocation Lists

Most traditional approaches to revocation require the party verifying the credential, called the **verifier** or **relying party**, to check back directly with the issuer or some other central authority to see if the credential is still valid. This approach has three major downsides:

1. It places a large technical burden on the credential issuer, who needs to create and maintain an API to provide access to the credential revocation list and provide permission to all those relying parties requiring access to it.
2. It requires the relying party to create and maintain calls to all those APIs from every credential issuer around the world.
3. It is a massive privacy issue because it provides an ideal way for issuers and relying parties to correlate an identity holder's usage of a credential across domains.

To take one well-known example of these issues, in order for a car rental agency to check if a UK driver's licence is valid, the driver must first, ahead of time, obtain a "check code" from the UK driving licence authority (the DVLA). To obtain this check code the driver must provide personal information including their national insurance number and postcode.

Immediately this creates a privacy risk—the DVLA warns of data "leakage" that occurs as a result of this transaction, stating "details from your DVLA record and your National Insurance number will be shared with other government departments (HMRC and DWP) to check your identity".

The DVLA then provides the driver with a check code which is valid for 21 days, which the driver must then provide to the car rental company. The car rental company uses the check code as user authorisation to execute a query on the DVLA's database. This complex

interaction is necessary because the DVLA cannot permit unrestricted access to its database without driver consent.

Whatsmore, throughout this process the DVLA is able to see and correlate a given driver's activities, which may be undesirable to the identity holder.

The Sovrin Solution: Privacy-Preserving Revocation Registries

To eliminate all of these issues, Sovrin developed a powerful new solution to credential revocation that is decentralized, asynchronous, and private. It's called **revocation registries**.

A revocation registry is data structure written to the Sovrin ledger by the issuer. It references the credential definition and contains a single (long) number called a **cryptographic accumulator**. This number can be checked instantly by any relying party when it needs to ensure a data in a proof it has been given hasn't been revoked by the issuer. It uses zero-knowledge cryptography to prove set membership. You can think of it as a type of compound hashing function—the number's value changes when hashes of valid credentials are added to or removed from the list, but from the number itself it is impossible to know whether any particular credential is included in the list unless you are the credential holder.

Only the credential holder, using their knowledge of which credential belongs to them, can create a zero knowledge **proof of non-revocation**, i.e., a proof that their credential belongs to the set of valid credentials (without disclosing which one it is). A relying party that needs to know that a credential has not been revoked can use this proof of non-revocation, together with the cryptographic accumulator the issuer placed on the Sovrin ledger, to instantly determine whether the credential is still valid.

When an issuer needs to revoke a credential, all the issuer needs to do is "subtract" the credential hash from the cryptographic accumulator and post the new number to the Sovrin ledger. The moment that happens, the credential holder will no longer be able to produce a valid proof of non-revocation. The chronological ordering of the ledger ensures that a verifier always knows they are using the most recent accumulator.

Throughout this process:

1. The issuer does not have to maintain any public APIs or manage integration issuers with any relying parties.
2. Relying parties do not have to contact or integrate with issuers to determine the revocation status of a credential.
3. All of the privacy issues of cross-domain correlation between issuers and relying parties are avoided.

By removing all this complexity from the current models of data exchange, this new and highly efficient revocation mechanism can bring digital credentials into the Internet mainstream and help transform the personal data economy.

Agent Authorisation Policies

There is one more function that Sovrin identity holders need to protect their security: the ability to authorize their agent(s) on their different devices (phone, table, laptop, car, etc.) to present credential proofs on their behalf—and the be able to revoke that authorisation if a device is lost or compromised. For example, if an identity holder loses their mobile phone (an “edge agent”), they need to be able to revoke that edge agent from using their Sovrin wallet.

To do this, Sovrin has been designed to support **agent authorization policies**. This is another specialized use of cryptographic accumulators and zero-knowledge cryptography on the Sovrin ledger to enable an identity holder to prove to a relying party that a particular agent is authorized to communicate on the identity holder’s behalf.

When a Sovrin identity holder authorizes a new agent to act on the identity holder’s behalf (e.g., buys a new phone or tablet), the identity holder adds an authorization key to the agent authorization policy registry. Then, when the agent opens a connection with a relying party, it signs its messages with its authorization key, and the relying party checks the agent authorization policy registry to be sure the authorization key has not been revoked.

If the device is lost or compromise, the identity holder de-authorizes the agent by removing the authorization key from the agent authorization policy registry. Now any relying party will know that the agent is no longer authorized by the identity holder.

Note that this functionality is not yet built in the open source Hyperledger Indy code, but is being designed at the time of writing.

No Private Credentials on Sovrin

To reiterate the point for emphasis, Evernym and the Sovrin Foundation have designed Sovrin so that private credentials and private DIDs are not stored on the ledger. The philosophy is one of privacy-by-design and privacy-by-default.

First, all cryptography has a shelf-life, and so it is inevitable that at some point that encryption will be broken.

Second, should there be any compromise of your keys, or if an issuer or relying party with whom you have shared this encrypted data is compromised, an attacker may be able to retrieve this record of your data which you can never deny existed and can never delete.

Thirdly, blockchains and distributed ledgers are immutable, permanent record of transactions. Storing private information on such a medium creates an indelible record that can be correlated even if the information is never decrypted. This is not privacy enhancing.

Evernym's recommendation is therefore is to keep private data, including encrypted private data or hashes of private data, off of any ledger than is publicly accessible.

Summary

This paper has described the policies for the data that does and does not go on Sovrin, and the logic behind those policies.

Using Sovrin and the mechanisms described in this paper, a relying party can, for the first time, do all of the following without ever needing to contact the issuer of a credential:

- Confirm that the data provided to them (or proofs about such data) by an identity holder came from the stated issuer;
- That the data is unchanged;
- That the data was provided only to the identity holder who has presented it;
- That the data has not been revoked by the issuer;

For an issuer, the consequences are similarly profound and liberating. An issuer can:

- Create and issue any type of credential without waiting for a central body or data hub to update its limited transaction set to accommodate it;
- Revoke credentials that it issues, without needing to create complex and privacy-leaking user experiences or handle multiple technical integrations with thousands or millions of relying parties;
- Provide its customers with trustworthy digital credentials that can be used anywhere in the world instantly.
- Revoke the ability of a lost/stolen device to be used to fraudulently represent the identity holder.

And all of the above can be done in a highly privacy preserving manner, protecting the identity holder from intended or unintended correlation and inadvertent data leakage.

Taken together we believe represents a revolutionary way to manage and share private and personal data—one that will gradually remove the need for old-style hub-and-spoke attribute exchanges and replace it with private, peer-to-peer exchanges of verifiable credentials.

In summary:

Goes on Sovrin

- Public DIDs and associated DID documents with verification keys and endpoints.
- Schemas and credential definitions
- Revocation registries
- Agent authorisation policies

Does not go on Sovrin

❌ Private DIDs
❌ Private credentials
❌ Consent receipts or data exchange records