2020-11-10 Indy DID Method Specification Call

Summary

- Update from DID F2F last week – resolutions?
- Continued: About the <network> element of the DID

Hyperledger is committed to creating a safe and welcoming community for all. For more information please visit the Hyperledger Code of Conduct.

Antitrust Policy Notice

Linux Foundation meetings involve participation by industry competitors, and it is the intention of the Linux Foundation to conduct all of its activities in accordance with applicable antitrust and competition laws. It is therefore extremely important that attendees adhere to meeting agendas, and be aware of, and not participate in, any activities that are prohibited under applicable US state, federal or foreign antitrust and competition laws.

Examples of types of actions that are prohibited at Linux Foundation meetings and in connection with Linux Foundation activities are described in the Linux Foundation Antitrust Policy available at http://www.linuxfoundation.org/antitrust-policy. If you have questions about these matters, please contact your company counsel, or if you are a member of the Linux Foundation, feel free to contact Andrew Updegrove of the firm of Gesmer Updegrove LLP, which provides legal counsel to the Linux Foundation.

HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Recording from the call: 20201110 did indy Method Specification Call Recording

Welcome and Introductions

Announcements

- TBD

Collaboration Channels

- Current hackmd document
- indy-did-method on RocketChat - https://chat.hyperledger.org/channel/indy-did-method
- indy-did-method repo
    - ReSpec vs. SpecUp

Discussion

- Update from the recent W3C DID Core Face to Face sessions - Brent Zundel
    - `type` will not be included in the spec, which impacts how we will store non-DID ledger objects (e.g. schema, etc.) as DIDs
    - JSON, JSON-LD and CBOR will all be supported in the spec., although DID Methods have the option of supporting some representations
        - This impacts the `type` issue in that we can solve the issue in just, say JSON-LD, and cover it in the other representations.
- `The <network> element of the DID – did:indy:<network>:<id>`

- Proposals to start the meeting:
  - Hash (did:indy:543F4:<id>) – unrecognizable, verifiable with the ledger, short, non-discoverable/requires a registry
  - Domain Name (did:indy:example.com:<id>) – recognizable, discoverable, not tied to the ledger, dependent on DNS
  - Arbitrary Name (did:indy:Sovrin:<id>) - recognizable, non-discoverable/requires a registry, not tied to the ledger
  - Hash plus an alias based on the domain name (this is what TrustBloc does)
  - Combination of arbitrary name and hash e.g. did:indy:sovrin:543F4:<id>

Approach	Discoverability	Decentralized Naming	(Limited) Verifiability	Human Friendly	Conciseness	Dependencies
~~Hash of Domain Genesis File~~	~~No~~	~~Yes~~	~~Yes~~	~~No~~	~~Yes~~	~~Registry or Config~~
~~Domain Name~~	~~Yes~~	~~Yes~~	~~No~~	~~Yes~~	~~No~~	~~DNS~~
Arbitrary Name	No	No (collision risk)	No	Yes	Maybe	Registry or Config
~~Hash and Domain Name Alias, as in TrustBloc~~	~~Yes and No~~	~~Yes~~	~~Yes~~	~~Yes and No~~	~~Yes and No~~	~~DNS and Config~~
~~Arbitrary Name + Hash~~	~~No~~	~~Yes~~	~~Yes~~	~~Yes~~	~~No~~	~~Registry or Config~~

- Online discussion after last week's meeting – about the value of the hash for verifiability and the risk of spoofing.
  - Purpose of the hash is not so much for verifiability as for decentralized naming – no risk of squatting on the same name as there would be with an arbitrary name.
  - Added the "Decentralized Naming" to the table above.
- The in-call discussion about the proposals:
  - Eliminated the DNS-based one as not desirable because of dependency on DNS
  - Eliminated the hash and DNS-based alias because of the complexity in using the two names for the identifier in signing scenarios, and the use of DNS.
  - Eliminated the hash only approach because of the lack of benefit of doing the hash vs. the downsides
    - The verifiability in using the hash is lost by using just 5 characters, but using a sufficient number makes the identifier too long
      - Micha generated a new Sovrin genesis file with a matching hash by varying only the alias name of one of the nodes in 30 minutes
      - Since we assume there will be in the low thousands of networks at the outside, the decentralization inherent in using the hash is not crucial
  - Eliminated the hash + arbitrary name because the hash is not providing additional value.
  - Leaving "Arbitrary Name" as the selected solution.
- Next questions:
  - Third level names for subsidiary ledgers – e.g. Sovrin Staging and Sovrin Builder net?
  - How to find the nodes of the network once the ledger is known? Config files, registries, gossiped names, etc.