JOLOCOM

Own your digital self

JOLOCOM.IO

Jolocom GmbH
Prinzessinnenstr. 19-20, 10969 Berlin, Germany

Self-Sovereign and Decentralised Identity By Design

Charleen Fei, Joachim Lohkamp, Eugeniu Rusu,
Kasia Szawan, Kai Wagner, Natascha Wittenberg

March 9, 2018

Contents

1 Vision 3

2 Design Decision Factors 4

3 System Architecture 5

3.1 Identity Generation Workflow 6

3.2 Hierarchical Deterministic Key Derivation 7

3.3 Decentralized Identifiers and DID Documents 8

3.4 Verified Credential Workflow 9

4 Conclusion 10

1 Vision

The originating vision of the Internet was that of a cooperative space, a non-proprietary community linked together in a network of distributed nodes to create a universal system of communication. It is now an unfortunate reality that a handful of proprietary companies dominate the digital landscape. Altogether, the services that they offer collectively comprise a vast percentage of total user activity on the web. To truly re-decentralise this feudal landscape, we must first start with the decentralisation of the core infrastructure of our digital life: identity.

With this background in mind, Jolocom is developing a solution which provides users with a decentralised identity based on hierarchically deterministic keys (HD keys) generated, provisioned and controlled by the users themselves. The idea of using public/private key infrastructure to manage identities is not a new one (see, Web of Trust, DPKI). However, key management and recovery in a decentralized approach to identity has largely been inefficient and resource intensive, with many approaches simply offloading complexity related to these issues onto the end user.

An identity based on HD keys allows for the easy management of multiple 'personas' and preservation of pairwise anonymity in context-specific interactions, as well as recovery of all of these derived key pairs with a simple seed phrase. Furthermore, our solution enables the modelling of ownership of IoT devices for integrated human/machine identity through the provisioning of HD child keys, providing Jolocom identity users a way to construct a full-fledged, decentralised digital identity which truly reflects all of their attributes - a truly self-sovereign decentralised identity.

2 Design Decision Factors

Usability is key to enable self-sovereign identity for people with little technical background. Our design decisions are thus made with the objective of a frictionless user experience, with core features like a user-centric single-sign on solution (SSO).

Security and secure communication in the form of pairwise anonymous communication must be supported, either using separate identities or discardable pseudonyms. We aim to abstract the necessary complexity of a secure, privacy-aware, and GDPR-compliant identity solution with the above described focus on frictionless user experience.

Integrated IoT and multi-faceted human identity refers to our model of identity as able to digitally reflect the multiple roles and personas that human identity necessarily contains, as well as the IoT devices which are an increasingly important aspect of our digital identities.

Portability of the identity created from our solution should be easy across multiple devices.

Compliance to existing industry standards and best practices concerning decentralised digital identities, specifically the W3C DID/DDO specification and the BIP 32/39/44 standards for the hierarchical derivation of Jolocom identity key pairs.

Maintenance of open source releases in order to support the larger decentralised application community.

3 System Architecture

The Jolocom library aims to simplify the management of identities and the data associated with them, unifying the underlying technologies into a single, developer-friendly RESTful API. Due to the rapidly changing decentralised technology landscape, the architecture is designed to not be bound to any particular technology, but instead to be able to adopt technologies as they arise that are most suitable to particular parts of the functionality the library exposes.

Currently, it exposes the following core identity management functionalities:

1. Generation of a unique, decentralised, and permanent global identity.

2. Derivation of child identities from this master identity to accurately model other personas and/or provision IoT devices.

3. Creation of verifiable claims associated with the identity which may be used in further interactions with services or other parties.

4. Association of created verifiable claims from third party with chosen identity.

User Interface - The user interface exists in an independent repository from the Jolocom Library, but consumes the endpoints exposed by the Library. It is the default user interface to create and manage identities and the claims associated with each identity, and allows for this management in a visual and user-friendly way.

Public Blockchain - This is the trusted storage layer for storing the mapping of each DID to their respective DDOs. We currently implement the Ethereum blockchain as our trust layer.

Storage Backend - This refers generically to any storage backend, public or private, and must not bound to any particular choice of technology. We currently use the Interplanetary File System (IPFS) with Interplanetary Linked Data (IPLD) as our default public storage backend for storing the DDO, which then contains content-addressed hashes for the specific retrieval of public verifiable claims. The default storage option for private claims will be directly on the personal device of the identity creator.

3.1 Identity Generation Workflow

The creation of an identity through the Jolocom library starts with encoded entropy which generates a BIP0032 conformant seed phrase. The derivation function may take any source of entropy, either user-generated or true entropy generated from a hardware source. From this seed phrase, a master key pair is derived. This master key pair will act as the parent key pair for all further derivation of child keys, and a compromise of the master key pair will result in a compromise of the user identity associated with this key pair, as well as all identity 'shards' derived from this key pair. For these security considerations, the master key pair will be directly encrypted after derivation and stored on the user's device.

Now, decentralised identity has been unlocked. With the master key pair, users may derive other key pairs which they control, but are not traceable to the master key pair. For instance, a generic signing key may be derived which the user may use to sign his or her verifiable claims. Or, in a situation requiring pairwise anonymity, a user may choose to derive a one-off signing key which, if compromised, can simply be discarded without consequence for the master identity.

3.2 Hierarchical Deterministic Key Derivation

The following sections describe how key generation and management is implemented technically.

Path Levels – A master key pair is derived from the seed phrase according to BIP0032. From this master key pair, child keypairs may be derived along different identity paths. An apostrophe indicates that BIP0032 hardened key derivation is used.

Identity path syntax: {m / purpose' / context' / entity'}

One key pair is derived per default in our implementation to make the identity ready to use. This key pair is defined with the following path:

Path definition: {m / 73' / signing-key' / global'}

Path implementation: {m / 73' / 0' / 0'}

Purpose – The purpose in the context of identity key generation is set as a constant to 73'. It describes that the keys under this node are all related to identity specific implementations. Hardened derivation is used here.

Context - The context field specifies the context in which the key is being used. Per default, only one context is defined which can be regarded as signing-key context. Customisation at this path depth allows for implementation of different personas and roles.

Entity - Entity defines the next path depth where keys can be further differentiated, e.g.:

{m / 73' / signing-key' / social-media'}

or:

{m / 73' / signing-key' / gaming'}

3.3 Decentralized Identifiers (DID) and DID Documents (DDO)

Although the HD keypair implementation above grants to Jolocom identity users the ability to model federated identities through parent and child key pairs, infrastructure based solely on public keys has historically been plagued with usability issues. Our solution concurrently implements the concept of decentralized identifiers (DIDs) and their corresponding DID document objects (DDOs) as described in the motivation behind the W3C DID Specification:

1. The new type of URL SHOULD NOT require a centralized authority to register, resolve, update, or revoke the identifier. The overwhelming majority of URIs today are based on DNS names or IP addresses that depend on centralized authorities for registration and ultimate control. DIDs can be created and managed without any such authority.

2. A URL whose ownership and associated metadata, including public keys, can be cryptographically verified. Authentication via DIDs and DID Documents leverage the same public/private key cryptography as distributed ledgers.

In the Jolocom implementation, a DID is generated from a user's public key. This DID resolves to a DDO stored on IPFS. The mapping of the DID to the returned IPFS hash will be stored as an entry in a registry smart contract on the Ethereum blockchain. As the storage layer for these mappings also acts as the trust layer, we have chosen the Ethereum blockchain for its data protection properties – data immutability, time stamping, and the possibility of public auditing.

The DDO is essentially a JSON object which describes the DID. This DDO will contain properties such as the DID which it describes, a collection of content-addressed hashes or other endpoints from which further identity related data such as verifiable claims may be fetched, as well as potentially equivalent DIDs (DIDs controlled by the same entity), authorization capabilities for entities to whom a user may delegate update control, or authentication credentials which may be used to authenticate as an equivalent DID.

3.4 Verified Credential Workflow

A verified credential is a simple file which contains a claim about an identity, e.g. "This user is employed by Company X", and a signature containing metadata about the claim created by the verifier identity, e.g. Company X. This credential is generated by the method contained in the Jolocom library, and will contain the id of the claim by which it can later be retrieved, the DID of the issuer, a timestamp of when it was issued, an expiry date for the claim, as well as the claim object itself containing the DID of the claims subject and the applicable claim, e.g.:

{did:jolo:0xd0ae58da9f72c48767b04f339a1a0142bb8e86b521d008ca65f7e3983b03d32b, ageOver: 21}

This entire credential object will be hashed and signed by the issuer, resulting in a two-part verified credential composed of the claim as detailed above, as well as a signature containing in its metadata the credential which has been signed. Accordingly, verifying a signed credential will also enable the party who is receiving the signed credential to check if the signature does indeed match what has been signed, ensuring for a secure and incorruptible verified credential.

Private verified credentials will be stored on the device or data storage option used by the party making use of the credential – in the example use case, the employee. These private verified credentials can be sent to a service or third party when the user decides to do so. Public verified credentials will be stored in the IPLD data structure. Initially, each DDO will contain a content-addressed IPLD hash to a credentials repository, which will serve as a head node from which further verified credentials will be linked and later retrieved. As verified credentials are received by each identity, these credentials will be stored as IPLD objects, and links – identified by the claim ID – will be created between each credential IPLD object and the initial credential repository found at the hash on the DDO. Therefore, all public verified credentials can be easily retrieved when needed by resolving the claim ID with the IPLD hash found in the identity DDO. As new links are added, the content-addressed hash of the credentials repository on the DDO will be updated accordingly, as well as the DID-DDO mapping on the Ethereum identity record for each identity.

4 Conclusion

We offer a truly self-sovereign decentralised digital identity solution which is part of Jolocom's vision to initiate a paradigm shift in modern information handling by putting the control of digital identity back into the hands of the users. Jolocom is an open-source project.