Technical specification

Version 0.2

1 Overview

Our system uses a blockchain to provide a decentralised record of individuals' professional credentials.

Individuals can ask organisations (or other individuals) to cryptographically attach references or endorsements to their profile. Requests for reference are implemented so that, once the reference is provided, the individual cannot claim it was unsolicited or refuse to accept it because of its content — references cannot be repudiated. Using the same process, organisations can request reviews.

When creating a new profile, users have to self-attach a reference that *grounds* the profile to a particular real-life identity (name, date of birth, corporate registration information, etc.). If a user wants to be able to give verifiable references, they must additionally have their identity grounded by a *trusted third party*.

The records attached to profiles, stored in the blockchain, do not include the actual content of the reference or endorsement, but merely a hash of the content. This also applies for the initial identity-grounding self-reference. Thus, profiles are public, but no data is disclosed by default. Users can choose which parts of their profile to disclose by selectively publishing the content of certain records.

2 Creating a profile and grounding its identity

To create a profile, a user must generate a public/private key-pair. The public key is hashed to obtain a profile *address*, which is used to refer to the profile. The private key is used to control the profile.

Before being able to request references from others, a user has to self-attach a reference that grounds their identity. This can be done *only once*.

Field	Value	Salt	Hash
Name	Michael Longworth	@38Bv>y+ve5RiVwZ	9916ffbd3944e34fb6d62e1048571e100b47ab5152fe53d21f261ff7d9b2d566
Date of birth	1980-12-02	6W3aP5d!ZnR	c300d1a45fb137bdcec8c3171a75949278c7b400b6a16df986315700380892183
Identification code	61152345	VWn9fpQ(5LHzZ	633209a51879bb3b64293c0ac16e72770ea55c8d5c1013febbf4802b4fdc3c2b

To self-attach a reference that grounds your identity, proceed as follows:

1. For each (`fieldName`, `value`) pair, generate a unique random `salt`

2. Hash each (`value`, `salt`) tuple separately
3. Collate all (`fieldName`, `hash`) pairs
4. Sign the result with your private key to obtain your *grounded identity*
5. Self-attach your grounded identity to your profile

The user's profile management software needs to store the field `value`s and `salt`s to be able to, at a later time, prove to a third party that the profile is indeed linked with the user's real-life identity.

2.1 Third-party grounding

If a user (organisation or individual) wants to give out references that are verifiable by others, in addition to the procedure above, their identity must also be grounded by a trusted *identity provider* (IdP).

To have their identity grounded by an IdP, users must provide proof of their real identity. For example, they could provide a copy of their passport or corporate registration documents.

For easier use in practice, the profile management software can be shipped with a set of predefined trusted IdPs.

2.2 Changing identities

If a user wants to change a part of their attached identity (e.g. they have changed their surname), an IdP has to check that the old and new identities refer to the same person.

3 Attaching records to profiles

Records are implemented as smart contracts. To request a reference or endorsement, a user (**A**) creates a smart contract that the referee (**B**) can add to. For example:

Field	**Initial value**	**Comment**
belongsTo	$addr_A$	Immutable
providedBy	$addr_B$	Immutable
referenceHash		
Method	**Can be called by**	**Comment**
addReference(*hash*)	B	Can only be called once

To provide the reference, **B** calls the `addReference()` method of the contract and passes it a hash of the reference's content. Through external means (e.g. via e-mail), **B** sends **A** the content of the reference and discloses a part of their identity (e.g., `value` and `salt` of their name). **A** stores this information for later use.

The system is extensible, so different smart contracts can be deployed for different types of records: certificates, employment records, professional association membership, school transcripts, etc.

3.1 Floating records

A *floating record* is attached directly to a real-life identity, rather than to a profile. Using floating records, organisations can emit certificates for people who haven't yet created a profile.

To claim a floating record, a user has to prove that their profile's grounded identity is the same as the record's.

4 Publishing verifiable CVs

4.1 Publishing a single verifiable reference

To make a reference provided by **B** verifiable, **A** discloses their address, the reference's content (e.g. *Good employee*) and part of **B**'s identity (e.g. *Name: ACME Corp*).

To verify the validity of the reference, proceed as follows:

1. Hash the reference content and check that it matches the stored `referenceHash`
2. Check that the smart contract `belongsTo` **A**
3. Verify that **A** disclosed **B**'s actual identity
4. Confirm that **B**'s identity has been grounded by a trusted IdP

If all the steps are successful, we have proved that **B** (*ACME Corp*) has said that whoever owns **A**'s profile was a *Good employee*.

Step 1 of the verification procedure may differ slightly based on the type of record that is being verified, as it depends on how the information is stored in that particular smart contract.

4.2 Publishing an entire verifiable CV

The procedure for publishing a single verifiable reference is *non-interactive*, i.e. it does not require a dialogue between **A** and the verifying third party. Thus, **A** can publish an entire verifiable CV by publishing all the constituent records individually and collating them in a single document.

A third party can verify the CV by validating all the constituent records and additionally checking that they all `belongTo` the same profile.