The latest generation
hybrid blockchain platform.

Secure identities in Shareable Private Vaults!

Concept

Version 0.X

DIDUX.IO

Smilo

Introduction

The team behind Smilo firmly believes that there has to be a connection between the blockchain and actual use cases. Currently, Smilo is the only applicable blockchain based computing platform for storing, sharing and revoking access to private data as an all-in one solution where the owner of the digital Identity is in full control on who, where and what is shared without revealing unnecessary assets.

New technology is always hard to implement in a correct way. And privacy often compromise user experience. With didux.io we take away these difficulties. The end user is in full control on who, where and what is shared with who, when and why.

This allows a quick and secure interaction between customers and companies, easy payments, access control, better service and much more.

Table of contents

Introduction 2

Table of contents 3

What is Smilo Platform? 4

- Introduction 4
- Specialties 4
- Unique strengths 4
- Products 4

Didux.io 5

- Introduction 5
- Problem 6
- Solution 7
- Unique features 8

Architecture 9

- Overview 9
- Digital Wallet (App) 9
- Smilo Blockchain 10
- Smilo Blackbox 10
- Customer Dashboard 11
- License Issuer 11

Technology 12

- Public/Private Keys Encryption 12
- Decentralised Identifiers 12
- Verifiable Credentials 14

Workflow End-user (App) 15

- Create Account 15
- Register/Fund account 15
- Create Decentralised Identifier 15
- Create Verifiable Credential 15
- Share Verifiable Credential 16
- Revoke Verifiable Credential 16

Credential Flow 17

Definitions 18

References and further reading 19

What is Smilo Platform?

Introduction

Smilo is a unique blockchain platform which supports the combination of hybrid transactions, hybrid smart contracts, and hybrid decentralised applications — with 'hybrid' referring to both public and private. Smilo's intent is to use blockchain technology to create an alternative protocol for decentralised applications.

Specialties

The Smilo team exists of a team of specialists in blockchain, decentralised storage, consensus mechanisms, smart contracts and decentralised identifiers. This includes private vaults, different encryption methods and digital identities with verifiable credentials.

Unique strengths

Smilo's unique strength comes from the combination of seven features:

- Smart contracts
- Decentralized applications
- Scalable
- Transparency
- Privacy and anonymity
- Connection to real life applications
- Open source

Products

Smilo comes with:

- Public or Enterprise blockchain
    - Mainnet
    - Testnet
- Blackbox for Private smart contracts
- Smart contract templates
- Web3 Libraries
    - Python
    - Java
    - JavaScript
- Block explorers
- Wallets
- Licensing model

Didux.io

Introduction

Didux.io means Decentralised IDentifiers User eXperience and is a Personal Information sharing and storing Platform. Where the user is in full control of their own personal data. The user has the power to create, verify and share their information, or subsets of it, with companies, agents and/or services based on advanced blockchain technology.

Problem

Organisations need a lawful basis for processing personal data. In the past, organisations relied on consent, but the GDPR toughened rules for obtaining and maintaining it. Means consent is always the least preferable option.

Of the remaining lawful bases, you would most likely be able to justify:

- Contractual obligations, if you need to process personal information to fulfil the terms of a contract (including an employment contract); or
- Legitimate interests, if there is any reason (including commercial benefit) to process personal information, provided it's not outweighed by negative effects to individuals' rights and freedoms.

However, these laws come with a cost. Organisations would be required to keep large volumes of sensitive information, which would create untold damage if breached. Given the current landscape of cyber crime, with organisations repeatedly breached despite adopting best practices, you might see it as simply too big a risk.

Individuals might also be hesitant about giving organisations their information. There is already growing concern over the way companies such as Facebook and Google use people's personal data, and if you are to persuade people to give you their personal data, you'll need to make a convincing argument about why it's necessary and how you will make sure it's kept securely.

Solution

Didux.io is built on top of the Smilo Blockchain which allows DID's to be hosted in a secure, immutable and decentralised way, without the need of a single centralised authority. Based on strong cryptographic proof, verifiable credentials (such as employee id's, birthdate, email address) can be created for each user. Since this is all private personal data, we share and process this in the ultimate secure way.

Contracts

Smilo Private Smart Contracts allows the user to setup a contract (mutual agreement) between 2 or more parties involved. This agreement contains the obligations, interests and personal data it involves. The Smilo Blackbox is the execution vault that secures the agreement and executes the private smart contract between the involved parties. Therefore execution, validation and data is only visible and secured to the parties involved.

The Private Smart contracts are protected by pointers on the public blockchain, creating a breadcrumb for the user without revealing any personal data.

Self sovereign

The user is in full control of the personal data involved. They are in control of sharing personal data and revoke access afterwards. This can even be automated by time slots in the private smart contract, allowing a company to access your personal data during working days, 8am-5pm, etc..

Secure sharing

Once you work in the blockchain space it is very easy to fall into the trap that all information should be public. Public information is important and has its place, but it is not suitable for everything. The number one fear people/implementers/regulators have about blockchain identity is if private information has to be stored on the public blockchain.

Currently, the only information Smilo requires to be stored publicly in the registry is a public key that can be used to verify signed data.

Most of the Credentials created about Decentralised Identities occur completely off the public blockchain. The signatures are verified through our registry of course, but the Credentials themselves don't need to ever hit the public blockchain.

By using the Smilo Blackbox, we share personal data in the utmost secure way. Contract sharing is done by a Peer-to-Peer encrypted communication.

GDPR Compliance

- The right to access – The user stores and owns the personal data. The companies does NOT store any personal data outside the blackbox.
- The right to be forgotten – The user can delete their personal data.
- The right to data portability – The user can transfer their data from one service provider to another and their personal devices.
- The right to be informed – The user has an access log where use of information is stored.
- The right to have information corrected – Users can update their own data.
- The right to restrict processing – Users can restrict that their data is not used for processing or revoke access to their personal data.
- The right to object – The contract protects unauthorized usage of information.
- The right to be notified – If there has been a data breach which compromises user personal data, the user is immediately noticed.

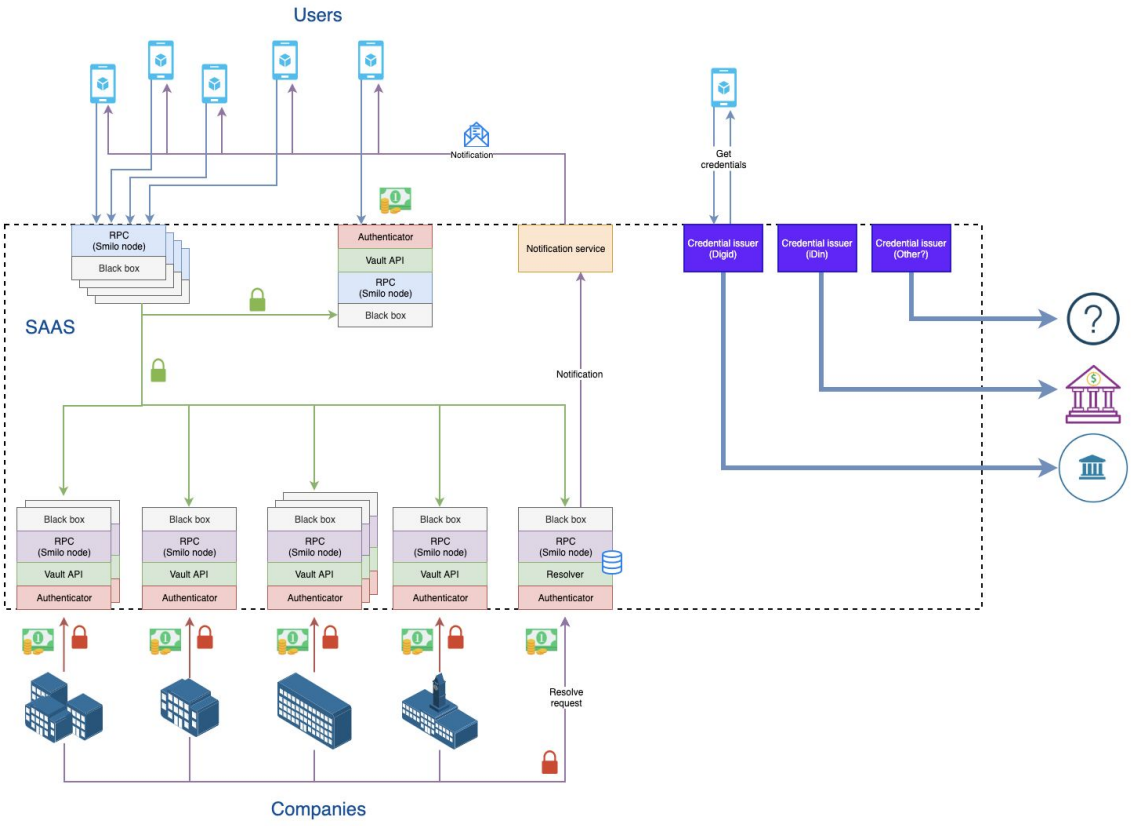
Unique features

Didux.io connects companies with customers in a secure and GDPR way.

- Secure vaults for all end-users
- Optional notification service for registered users
- End-user is in full-control
- Encrypted data sharing between 2 or more parties
- Didux.io can not access any data
- Vault can execute private smart contracts
- Multiple possibilities for issuing verifiable credentials
    - Digid
    - iDin
    - Custom companies

Architecture

Overview



Digital Wallet (App)

The digital wallet, also known as the Didux.io app, stores all the user credentials and personal data. Secured by the users keys and a password, the phone can lock/unlock all credentials and share these with other parties.
Credentials can be, but are not limited to:

- Full name
- Age
- Customer ID
- Employer ID
- Passport
- Driving license
- DID
- School/University Grades

Smilo Blockchain

The Smilo blockchain is a public blockchain used to create and store the Decentralised Identifier (Did), including the authorisation keys used for the identifier.
Smilo public Smart contracts are designed with Compliance by Design and Trust by Design. The decentralised and immutable character of the blockchain ensures the security of the Did.

The blockchain itself does not store any personal data.

Smilo Blackbox

The Smilo Blackbox, also known as Smilo Private Vault, is designed for Privacy by Design, extending the Smilo Public Blockchain.

The Smilo Private Vault (SPV) is P2P software that stores and shares your private smart contract state across the network. It is a software program written in Golang that makes use of the Ethereum P2P capabilities to find peers and transfer encrypted data.

The Smilo Private Vault is created to encrypt and share private transactions via P2P, outside of the blockchain. The protocol will search the network and identify the Vault peer that matches with the peer that is supposed to receive the data at the time of creating the transaction. During this transaction, the smart contract state will never be saved on the blockchain, only a hash that is consequently used as a checksum will be stored on the blockchain. With this hash, nodes can validate whether they have received a valid state or not. A not valid state is automatically refused and the peer who sent the faulty data is blacklisted temporarily.

Public smart contracts cannot interact with Private smart contracts, as private smart contracts are only known to certain entities. However a private smart contract can make use of a public smart contracts information to change its own state. But will never be able to change the Public smart contract's state. The data that is sent via P2P to a node, is encrypted using TweetNACL "crypto library in a 100 tweets" in Golang.

Private Vaults are completely GDPR compliant and developed with of the Compliance by Design and allows the owner to completely remove the vault, without any traces left.

Customer Dashboard

The Customer dashboard allows customers to add/remove/edit/monitor instances in the infrastructure and manage their payments.

Basic functionality:

- Deploy contracts
- Read contracts
- Update contracts
- Generate pub/private keys
- Key management
- Create Credential requests (including QR-codes)
- Verify and Sign Credentials

License Issuer

The license issuer is a generic faucet, used to reward 'new' identities with an x amount of Smilo. Smilo is needed to deploy (private) smart contracts on the decentralised blockchain.

The license issuer can be managed by the customer our any other party that wants to reward people that register a Decentralised Identifier.

Technology

Public/Private Keys Encryption

Public key cryptography uses a pair of a public key and a private key to perform different tasks. Public keys are widely distributed, while private keys are kept secret.

Using a users public key, it is possible to encrypt a message so that only the person with the private key can decrypt and read it. Using a private key, a digital signature can be created so that anyone with the corresponding public key can verify that the message was created by the owner of the private key and was not modified since.

The Smilo Blockchain makes extensive use of public key cryptography based on elliptical curve ecp256k1.

Public keys are sized so that, absent a breakthrough in solving the discrete logarithm problem, brute force is impractical for any conceivable amount of computing power. With 2^{128} possible combinations, and if we assume a modern computer can compute a billion per second (which is a massive overestimate), it would take a million such computers ~5e15 years to brute force your key. If we assume computing power improves by another million fold, it'd take this massive cluster roughly about 5 billion years.

Decentralised Identifiers

Decentralised Identifiers (Did) are the roots of an Identity. Simply said, it is the Universally Unique Identifier (UUID) used to manage all verifiable Credentials, and does not contain any personal data.

```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:smilo:123456789abcdefghi",
  "authentication": [
    {
      "id": "did:smilo:123456789abcdefghi#keys-1",
      "type": "Secp256k1VerificationKey2018",
      "controller": "did:smilo:123456789abcdefghi",
      "publicKeyHex": "0x699746b47142727e6dc5e939a9e26803865456b5"
    }
  ],
  "created": "2018-10-10T17:00:00Z",
  "updated": "2019-01-10T17:00:00Z"
}
```

Example: A minimal Decentralised Identity.

Goal	Description
Decentralization	DID architecture should eliminate the requirement for centralized authorities or single points of failure in identifier management, including the registration of globally unique identifiers, public verification keys, service endpoints, and other metadata.
Self-Sovereignty	DID architecture should give entities, both human and non-human, the power to directly own and control their digital identifiers without the need to rely on external authorities.
Privacy	DID architecture should enable entities to control the privacy of their information, including minimal, selective, and progressive disclosure of attributes or other data.
Security	DID architecture should enable sufficient security for relying parties to depend on DID Documents for their required level of assurance.
Proof-based	DID architecture should enable the DID subject to provide cryptographic proof of authentication and proof of authorization rights.
Discoverability	DID architecture should make it possible for entities to discover DIDs for other entities to learn more about or interact with those entities.
Interoperability	DID architecture should use interoperable standards so DID infrastructure can make use of existing tools and software libraries designed for interoperability.
Portability	DID architecture should be system and network-independent and enable entities to use their digital identifiers with any system that supports DIDs and DID Methods.
Simplicity	To meet these design goals, DID architecture should be (to paraphrase Albert Einstein) "as simple as possible but no simpler".
Extensibility	When possible, DID architecture should enable extensibility provided it does not greatly hinder interoperability, portability, or simplicity.

Verifiable Credentials

A verifiable credential can represent all of the same information that a physical credential represents. The addition of technologies, such as digital signatures, makes verifiable credentials more tamper-evident and more trustworthy than their physical counterparts.

Depending on the agent and their request, an identity can share a verifiable Credential containing the proof for the requested credentials. The credentials are signed by the DID keys to proof origin and protect against abuse.


```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "type": "VerifiablePresentation",
  "verifiableCredential": [
    {
      "id": "http://example.edu/credentials/1872",
      "type": [
        "VerifiableCredential",
        "FaceBiometricsCredential"
      ],
      "issuer": "https://example.edu/issuers/565049",
      "issuanceDate": "2010-01-01T19:73:24Z",
      "credentialSubject": {
        "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
        "faceVectors": "0.001, 0.001, ....."
      },
      "proof": {
        "type": "RsaSignature2018",
        "created": "2017-06-18T21:19:10Z",
        "creator": "https://example.edu/issuers/keys/1",
        "jws": "eyJhbGciOiJSUzI1NiIsImI2NCI6ZmFsc2UsImNyaXQiOlsiYjY0Il19..TCYt5"
      }
    }
  ],
  "proof": {
    "type": "RsaSignature2018",
    "created": "2018-09-14T21:19:10Z",
    "creator": "did:example:ebfeb1f712ebc6f1c276e12ec21#keys-1",
    "nonce": "1f44d55f-f161-4938-a659-f8026467f126",
    "domain": "4jt78h47fh47",
    "jws": "eyJhbGciOiJSUzI1NiIsImI2NCI6ZmFsc2UsImNyaXQiOlsiYjY0Il19..kTCYt6"
  }
}
```


Example: Verifiable Credential

Workflow End-user (App)

Create Account

When a user opens the app, the user is prompted by a screen where he can create an account or import a backup. The account created is a local account, used to store Credentials and keys. Each user device can have their own account.

Register/Fund account

Before a user can deploy a Decentralised Identifier, the user need to be part of the Smilo Infrastructure. There are 3 ways to become part of the infrastructure:

- Buy Smilo from an exchange
- Register at the License Issuer
- Fund the account with already owned Smilo

Buying Smilo at an Exchange can be done anonymous, but can be hard for 'regular' users.

Registration at the License Issuer can be done once, by verification over email. When the user registers himself, he will be rewarded with x Smilo. More than enough for maintaining his DID and Claims. This option is preferred, and easiest in user experience.
License issuers should be seeded with funds from the requesting companies, since they benefit from the user to enroll himself in the platform.

When the users already have Smilo available, he can send x Smilo to the newly created account to deploy and maintain his DID.

Create Decentralised Identifier

When an account is funded, the user is able to deploy a Decentralised Identifier.
The DID contains the keys of all accounts connected to this DID. It is possible to have multiple accounts on different devices (Phone, laptop, tablet) all connected to one DID. When a phone or tablet is lost, it is possible to revoke keys and add a new account.

Create Verifiable Credential

When the DID is deployed, we can add multiple Credentials in the app.
Credentials can be, but are not limited to:

- Full name
- Age
- Customer ID

Some Credentials need third party verification, like a passport, age or customer ID. Depending on the credential we can request a verification of a credential to a third party, like DigiD and iDin.

All Credentials are stored on the phone, encrypted with a password, and only accessible by the user. There are possibilities to sync Credentials between phone, tablet and/or computer, or use encrypted cloud storage for backup.

Share Verifiable Credential

When a company would like to know additional information, we receive a share request. This can be, but is not limited to, a QR-code, email, webpage, etc.

Scanning or clicking the request will open the app en show a ‘subscription’ request.
The subscription request contains:

- Who request Credentials
- Which Credentials are requested
- Where the Credentials are used for

Missing credentials can be added by the user and the user can decide to Accept or Deny the request. By accepting, the user (app) create a Private smart contract which will be shared by the requesting party.

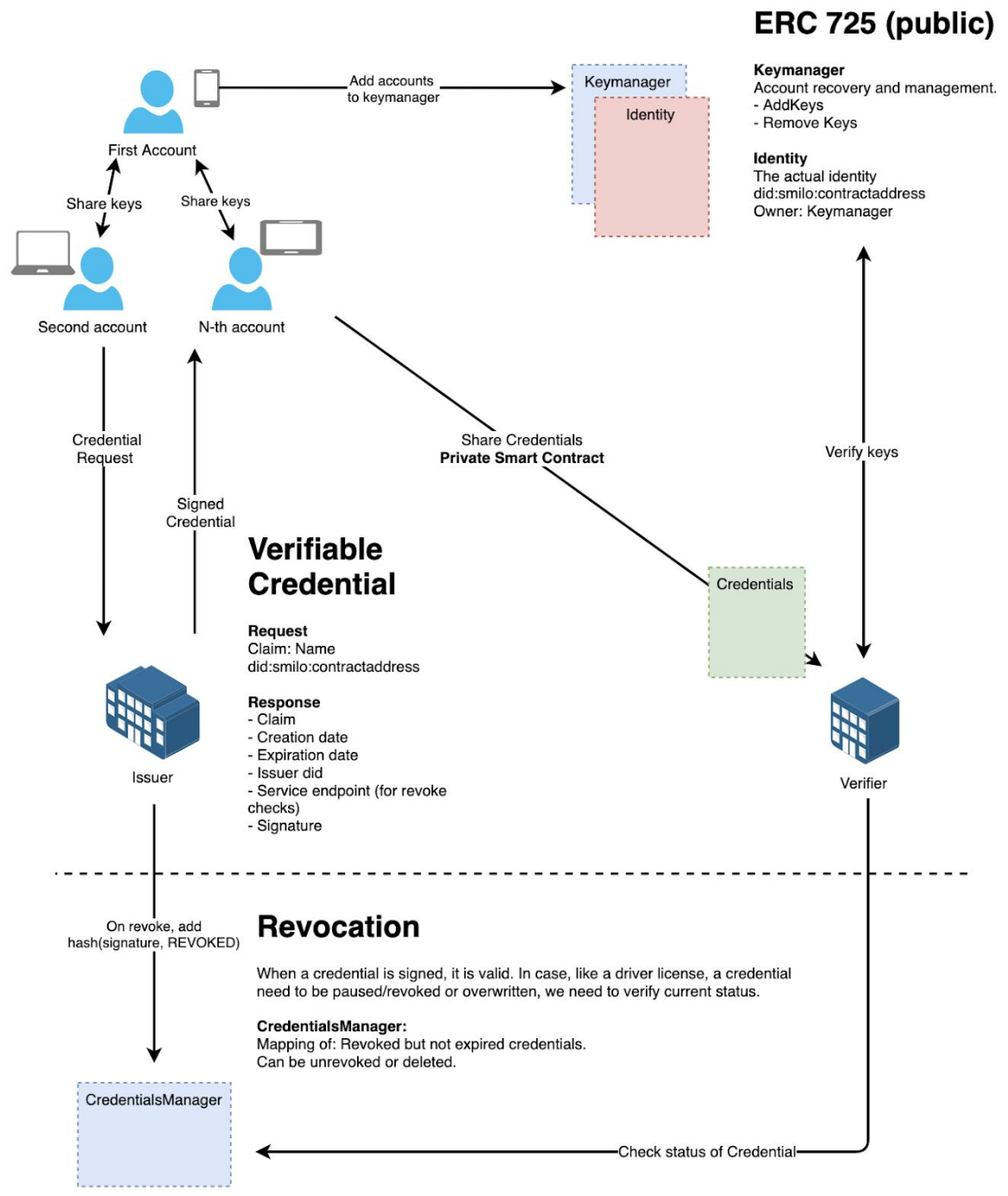
Revoke Verifiable Credential

When the identity want to revoke their data, they can open their app and open the active subscriptions. Active subscriptions can be edited, updated or deleted.

Add deletion, the Private smart contract between the 2 parties is revoked and destroyed.

The verifiable credentials are still available on the device, but the contract is ended and destroyed. The company receive an update on the contract and remove the record from the BIS.

Credential Flow



Definitions

User

A user is the owner of a digital Identity.

Processor

An identifier that interacts or processes users personal data.

Digital Identity

A digital identity is information on an entity used by computer systems to represent an external subject. That subject may be a user or a machine.

Digital Identifier

Digital identity fundamentally requires digital identifiers, strings or tokens that are unique within a given scope (globally or locally within a specific domain, community, directory, application, etc.). Identifiers are the key used by the parties to an identification relationship to agree on the entity being represented.

Self sovereign

Self sovereign is the concept of property in one's own person, expressed as the moral or natural right of a person to have bodily integrity and be the exclusive controller of one's own body and life.

Decentralised Identifiers (DID)

Decentralized Identifiers (DIDs) are a new type of identifier for verifiable, "self-sovereign" digital identity. DIDs are fully under the control of the DID subject, independent from any centralized registry, identity provider, or certificate authority.

Credentials

Credentials are object that proof our capabilities or claim of identification.Credentials are a part of our daily lives; driver's licenses are used to assert that we are capable of operating a motor vehicle, university degrees can be used to assert our level of education, and government-issued passports enable us to travel between countries.

Verifiable Credentials

A verifiable credential can represent all of the same information that a physical credential represents. The addition of technologies, such as digital signatures, makes verifiable credentials more tamper-evident and more trustworthy than their physical counterparts.

Standards

For the implementation of Smart Contracts, DID’s and Verifiable Credentials Smilo use the W3C Standards. The goal is to have contracts, DID’s and Credentials exchangeable with other DID-managers.

References and further reading

1. Smilo Website. https://www.smilo.io
2. Didux.io Website. https://www.didux.io
3. Smilo github. https://github.com/Smilo-platform
4. Smilo Mainnet Explorer. https://explorer.smilo.network
5. GDPR. https://gdpr-info.eu/
6. Decentralised Identifiers. https://w3c-ccg.github.io/did-spec/
7. Digital Identity. https://en.wikipedia.org/wiki/Digital_identity
8. Verifiable Credentials. https://www.w3.org/TR/verifiable-claims-data-model/
9. What is free software? https://www.gnu.org/philosophy/free-sw.en.html
10. GNU GPL V3 License. https://www.gnu.org/licenses/gpl-3.0.en.html