自己主権型アイデンティティに基づく ID 管理と利用の具体化

Self-sovereign identity management for Web3.0 services

トラストビルディング
木部　龍駿
指導教員： 細野　繁

東京工科大学コンピュータサイエンス学部コンピュータサイエンス学科
サービスシステムデザイン研究室

キーワード： Web3.0, ブロックチェーン, 自己主権型アイデンティティ

1　はじめに

現在, 実生活で年齢確認などの身分証明を行う際, 公的証明書を用いて目視確認を行っている. 証明書には個人情報が直接記載されている事があり, 悪用されるリスクが伴う. 同様に, デジタルアイデンティティにも漏洩リスクがある.Web2.0 で影響力や高いシェアを持つ企業が独占し, 中央集権型管理されていることから, サイバー攻撃等のリスクからも厳格に保護する必要がある. 個人情報を保護するため, 個人が管理主体でありデジタル空間以外でも適切な情報提示ができる管理方法の開発と実装が課題である.

2　自己主権型アイデンティティ

自己主権型アイデンティティ (Self-Sovereign Identity:SSI) とは, アイデンティティの管理主体が存在するのではなく, 個人は独立しており, 個人が自分自身のアイデンティティをコントロールできるようにすることを目指す思想である [1].

3　分散型 ID 管理

分散型 ID 管理では, 従来 Web2.0 で中央集権型管理されていた個人情報に対し（図 1）,Web3.0 では分散型管理へ移行（図 2）することで, ユーザーの個人情報が特定の ID プロバイダーに依存しないように, 分散台帳を用いることで依存度を下げることが可能な仕組みである.

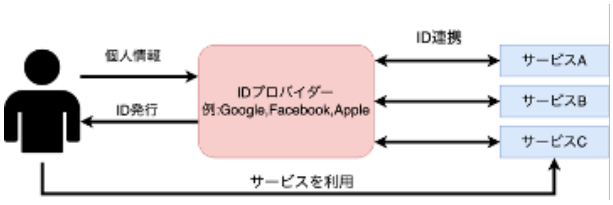

図 1　中央主権型アイデンティティ管理

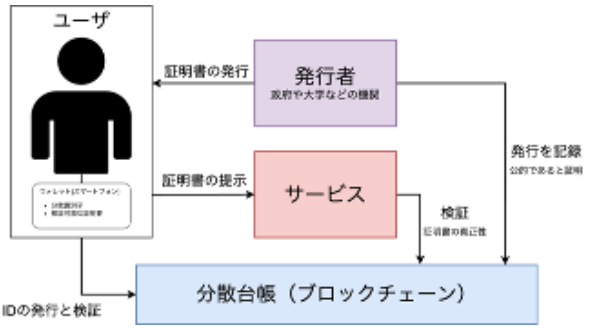

図 2　分散型アイデンティティ管理

4　ウォレットと証明書

ウォレットはデジタルな財布として, 分散台帳に登録する公開鍵のペアとなる秘密鍵と, 発行された証明書が保存されている.　証明書とは検証可能な資格証明書（Verifiable Credential：VC）のことを指している. 資格情報をデジタルでやりとりするためのデータであり, 政府などが検証することで, 公的であると証明することができる.

5　研究目的

本研究の目的は, 実生活で使われる運転免許証などの証明書は個人情報が直接記載されている事に対し, 分散型 ID 管理を利用したデジタル化を行うことで個人情報をユーザの管理下に置くこと目指す. また, デジタルな身分証明書として活用するために提示方法や検証方法を解く.

6　提案手法

6.1　従来型と非接触型での証明書提示の比較

従来は物理カードを提示することで身分証明を行っている（図 3）. 提案では, QR コード化した資格情報を提示し, サービス提供者が読み取ることで, デジタル証明書での身分証明を可能にする.

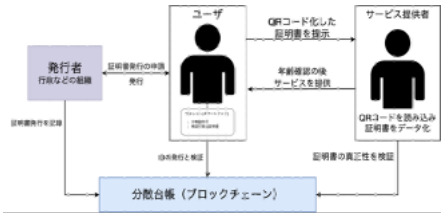

図 3　ユースケース

6.2　証明書発行の流れ

図 4 で証明書発行の流れを示す．自己主権型の概念を元に, ユーザ主体での証明書発行を行う. 発行された証明書はウォレットアプリへ保存される.

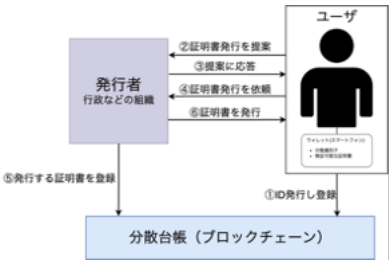

図 4　証明書発行の流れ

6.3　検証の流れ

図 5 で証明書検証の流れを示す. 証明書を QR コード化し, サービス提供者は読み込むことで検証を行う. 検証にはゼロ知識証明が使われ, 秘密の情報自体を明かさずに相手に証明する事ができる.

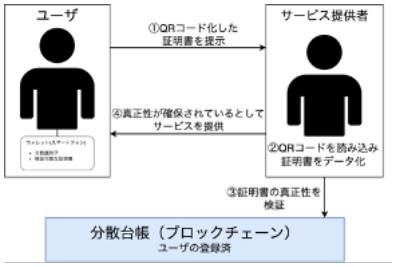

図 5　証明書検証の流れ

7　評価

7.1　実装環境

図 3 ユースケースを基に実装環境を下記の表 1 に示す.

表 1　実装環境

OS	Ubuntu Server20.04
アプリケーション	React Native
分散型 ID	Hyperledger Indy
Agent Framework	Hyperledger Aries

7.2　構成

構成に利用した技術及びアーキテクチャを図 6 としてまとめる.Hyperledger プロジェクト [2] から, 分散台帳を Indy で構築し,Aries では発行者となる Issuer, サービス提供者となる Verifier, ユーザとなる Holder-App を構築する. 各エージェントがコネクションを確立することで, 分散型 ID 管理システムを構築することができる.

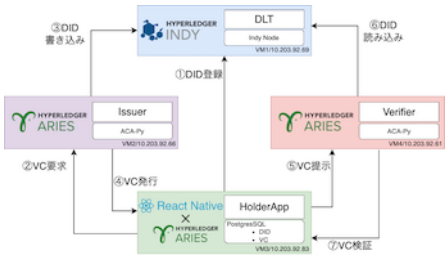

図 6　アーキテクチャ

ユーザが実際に利用する HolderApp は,Aries が提供する JavaScriptFramework を活用し,ReactNative でのアプリケーション実装を行う. 実装には,React-native-qrcode-svg package を使うことで json 形式の証明書から QR コードの発行を行う. また, 証明書検証を行う Verifier でもアプリを作成し,QR コードを読み込み証明書の検証を行う.

8　考察

既存の公的証明書に分散型 ID 管理を軸としたデジタル化を行うことで, 従来よりセキュアな証明が可能となる. また,QR コードでの証明に求められることとして, 提示から検証結果までどの程度の時間を必要とし, 一連の提示の流れで利便性を高めるために時間が鍵になると考えた.

9　今後の計画

QR コード発行と QR コードを読み込み検証を行うアプリの開発を行う. また,QR コード生成にかかる時間, 検証にかかる時間の統計をとる.

参考文献

[1] Sovrin Foundation, Principles Of SSI V3, 2020, https://sovrin.org/principles-of-ssi, 閲覧日 2022-10-15

[2] Hyperledger Foundation, Tutorials, 2015, https://www.hyperledger.org/use/tutorials, 閲覧日 2022-10-15