

マイナンバーカード, BBS署名

光成滋生

last update: 2025/11/06

目的

- マイナンバーカードの仕組みと課題を理解する
- SD-JWTやBBS署名などの選択開示署名の概念を理解する

用語一覧

- 本人確認
- マイナンバーカード
- 選択的開示可能署名
- SD-JWT, BBS署名, グループ署名

本人確認ガイドライン

本人確認 = 身元確認 + 当人認証

- 身元確認 = 登録する基本4情報などが正しいことを確認する
 - L3: 対面で公的身分証を元に確認
 - L2: 郵送などの非対面で公的身分証を元に確認
 - L1: 自己申告
- 当人確認 = 登録された情報と、今回の情報を認証の3要素（生体・所持・知識）で照合
 - L3: 耐タンパ性ハードウェアを用いた複数の認証
 - L2: 3要素のうち複数用いる認証
 - L1: 3要素のうち1要素を用いる認証
- サービスレベルに応じてレベルは異なるが銀行などではL2+L2以上が多い
- 参考: IPA 「サービスに応じたデジタル本人確認ガイドラインの検討」

マイナンバーカード

公的個人認証サービス

- 署名用電子証明書
 - 「基本4情報（氏名・住所・生年月日・性別）+個人の検証鍵」にCAが署名したもの
 - マイナンバーカードは証明書を耐タンパー性のハードウェアに格納
 - いろいろなシーンで使ってほしい by デジタル庁
 - 利用者証明用電子証明書には基本4情報は含まれない

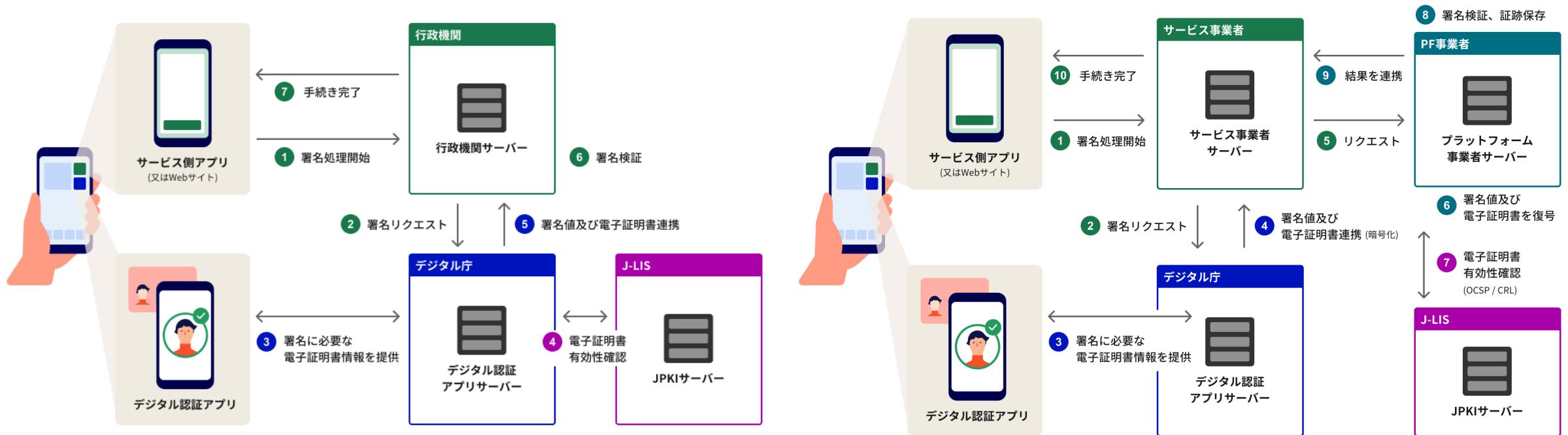
利用時の手順

1. なんらかの文書に署名鍵で署名する（要 署名用電子証明書のパスワード）
2. 「文書 + 署名 + 署名用電子証明書」を相手に送る
3. 相手はCAの検証鍵で署名用電子証明書の正しさを確認
4. 署名用電子証明書に含まれる検証鍵で文書の正しさを確認
ユーザの証明書が失効されていないかの検証も必要

署名の流れ

行政機関向け（左）と民間事業者向け（右）の違い

-



電子証明書の扱い

- 署名の検証に必要な電子証明書は直接民間事業者には渡さない
- CRL (証明書失効リスト) はシリアル番号と失効日時のリスト

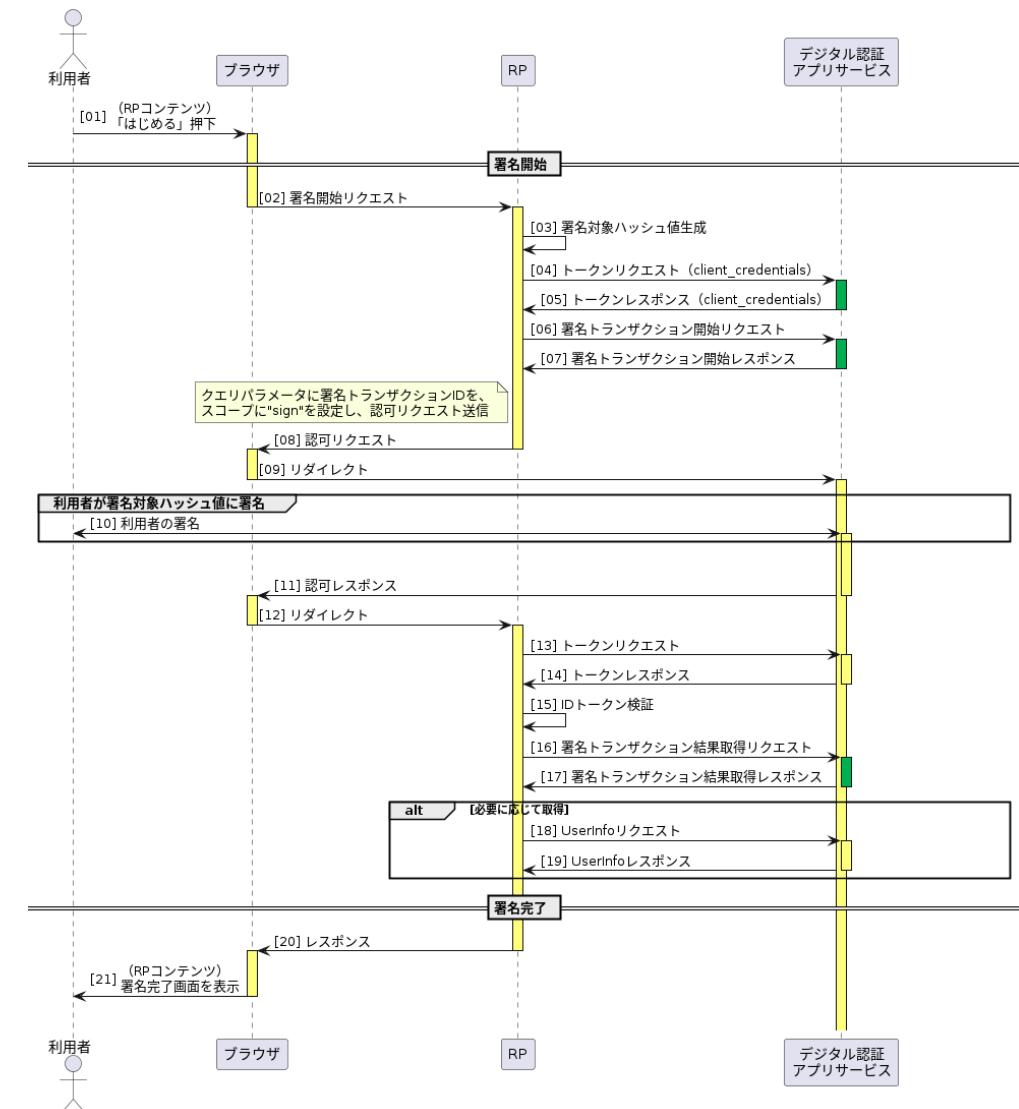
署名のシーケンス図

主な処理

- [10]でデータのハッシュ値に対して署名する
 - 署名用電子証明書の認証パス・有効期間の検証
 - 署名用電子証明書の有効性確認(行政機関のみ)

署名検証は行わない

- 署名対象データ・署名対象ハッシュ値・署名値の確認をしない
- 署名検証は自分（RPアプリ）でやる[17]
 - 実際にはPF（プラットフォーム）事業者に依頼する [JPKIサービス](#)



プライバシー

基本4情報は直接渡さない

- しかし代わりに絶対的な権限がデジタル認証アプリサーバーに集中する可能性
- 原理的には、いつ、だれが、どのサービスに何の情報を渡したのか把握できる

プライバシーポリシー

- 保有する情報を利用目的以外には使わない・提供しない
- 「情報の全部または一部の処理を第三者に委託する」場合もある

Apple Walletのマイナンバーカード

カード代替電磁的記録（属性証明機能）

- マイナンバーカードに記録された基本4情報（氏名・住所・生年月日・性別）
 - + その情報が正しいことを示す情報
 - + 送信者が本人であることを証明する情報
- 大多数のオンラインサービスでは十分な本人確認になる

iPhoneのVerify with Wallet API

- Apple Walletに格納された個人情報を扱うAPI
- 2025年9月対応しているのはUSの一部の州と日本のみ

仕様

- ISO/IEC 18015-5: モバイル運転免許証の規定
- ISO/IEC 23220: モバイルデバイス上で動作するデジタル身分証の規定

Apple Walletで確認できる項目

要求できる項目

- N歳（以上）であること
- ID写真
- 姓・名
- 住所
- 誕生日
- マイナンバー
- 性別

信頼性

- Appleとそのソフトウェアを信頼するしかない

選択的開示可能な署名

基本4情報（に限らないが）の一部のみを開示して確認できる暗号技術

- W3CのVC : Verifiable Credentials（検証可能な認証情報）などで規格策定中
 - 大きく二種類の方法
 - ハッシュ関数ベースのもの (SD-JWT)
 - ペアリング暗号ベースのもの (BBS署名)

メリットとデメリット

項目	SD-JWT	BBS署名
実装	容易	難しい
検証コスト	低い	大きい
unlinkability（後述）	弱い	強い

用語

- JOSE (JSON Object Signing and Encryption) : JSONを用いた署名と暗号化の規格
 - JSON + Base64URL
- JWS (JSON Web Signature) : 署名に関する規格
- JWE (JSON Web Encryption) : 暗号化に関する規格
- JWK (JSON Web Key) : 鍵に関する規格
- JWA (JSON Web Algorithms) : アルゴリズムに関する規格
- JWT (JSON Web Token) : クレーム（値）に関する規格
- SD-JWT ([Selective Disclosure for JWTs](#))
 - JWTで選択的開示を実現するための規格
- CBOR (Concise Binary Object Representation) : JSONのバイナリ版
 - COSE (CBOR Object Signing and Encryption) : CBORを用いた署名と暗号化の規格

SD-JWTのコアアイデア

salt+ハッシュ関数を使う

クレーム	クレーム値	salt (128bit)	ハッシュ値(_sd)
姓	山田	1aef...	H("山田", "1aef...")
名	太郎	584a...	H("太郎", "584a...")
誕生日	2000/01/01	39ff...	H("2000/01/01", "39ff...")

- クレーム値全体を署名するのではなくハッシュ値全体に対して信頼できる機関が署名する
 - ハッシュ値だけ公開する
- 検証方法
 - 検証者はハッシュ値を含む署名を検証
 - 開示したいクレーム値に対してsaltとクレーム値を要求し, ハッシュ値が正しいことを確認
- saltが128bitあるのでsaltが非公開だとハッシュ値からクレーム値を求められない

SD-JWTの暗号学的な欠点

署名は常に同じ

- いろんなサービスで同じSD-JWTを提示する

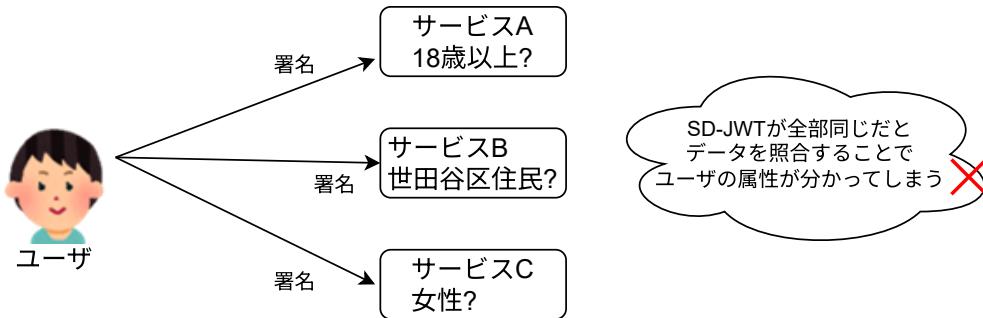
saltの漏洩

- クレーム値のドメインが小さいとsaltが漏洩するとハッシュ値からクレーム値を計算できる
 - $H(salt, \text{クレーム値}) = \text{hash 値}$ となるクレーム値の総当たりが可能
 - 性別なら精々数パターン, 誕生日も $365 * 100$ 程度

unlinkability（連結不可能性）とは

あちこちのサービスで利用したときに追跡されないこと

- 検証のために相手に渡すデータがいつも同じだと追跡される



- 3rd party cookieのようなもの
- プライバシーを保つにはunlinkabilityが重要

ペアリングベースの署名

ドラフト

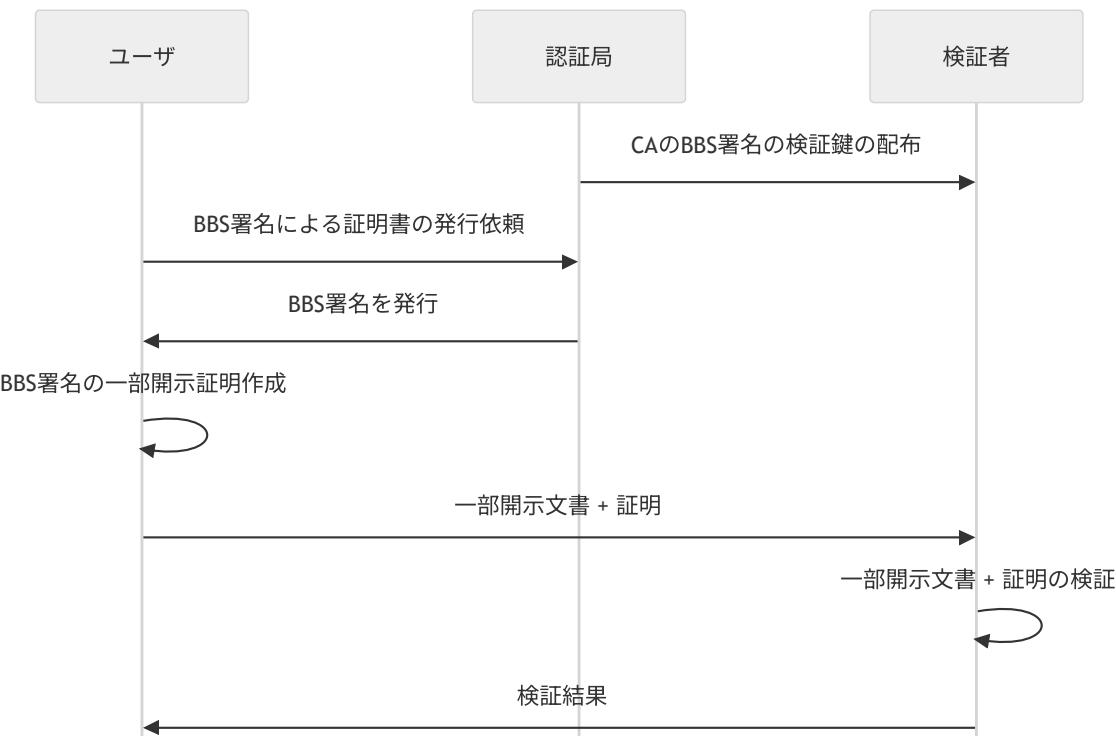
- Revisiting BBS Signatures, EUROCRYPT'23
 - [draft-irtf-cfrg-bbs-signatures](#)

フロー

- CAが基本4情報等に対してBBS署名を発行
- ユーザは開示情報を決めて「証明prf」を生成
- 検証者は開示情報とprfをCAの検証鍵で検証

SD-JWTに対する利点

- 同じ開示情報に対するprfでも毎回異なるのでunlinkabilityが高い



BBS署名のPoC

ブラウザで動作するデモ

- C++/LLVMで実装したものをWasm用にコンパイルしてNode.jsモジュール化
 - TypeScriptを利用してブラウザで動かす
- 細かいデータフォーマットなどは仕様に従ってないプロトタイプ実装

グループ署名

メンバーが自身の身元を明かさずに署名できる傘連判状のようなもの

基本アルゴリズム

- $\text{GKg}(1^\lambda) \rightarrow (gpk, ik, ok)$: グループ鍵を生成
 - gpk : グループ署名鍵 (group public key)
 - ik : 発行者秘密鍵 (issuer key), ok : 開示鍵 (opening key)
- $\text{UKg}(gpk, ik) \rightarrow gsk_i$: 発行者とメンバーの対話により各メンバーの署名鍵を発行
- $\text{Sign}(gpk, gsk_i, m) \rightarrow \sigma$: メンバーがメッセージ m に署名
- $\text{Verify}(gpk, m, \sigma) \rightarrow \{0, 1\}$: 署名の検証
- $\text{Open}(gpk, ok, m, \sigma) \rightarrow \{i, \perp\}$: 発行者が署名者の身元を開示

グループ署名特有の要件

- 匿名性: 発行者以外は署名者が誰か分からぬ
- 追跡可能性: 発行者は署名者を特定できる

BBS署名の原型

Boneh, Boyen, Shacham, "Short Group Signatures", CRYPTO'04

- $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$: 位数 r の加法巡回群
 $e : G_1 \times G_2 \rightarrow G_T = \langle g_T \rangle$: ペアリング
- アルゴリズム概要
 - 発行者は γ を生成しグループの鍵生成をした後 γ を破棄する
 - グループの公開鍵 gpk ($w = \gamma g_2$ を含む), 追跡用秘密鍵: $gmsk$
 - 各ユーザの秘密鍵 gsk_i ($x_i, \frac{1}{\gamma+x_i} g_1$)
 - 署名: $\text{sign}(gpk, gsk_i, \text{メッセージ } m)$ から署名 σ を作る
 - 検証: $\text{verify}(gpk, m, \sigma)$ で正しいか確認
 - 公開: $\text{open}(gpk, gmsk, m, \sigma)$ でメンバーの身元を明かす

応用

- 複数メッセージに対応し($\sum_i m_i h_i$)、その一部の秘匿をゼロ知識証明ZKPを使って実現 19 / 20

安全性の根拠

q-SDH (Strong Diffie-Hellman) 仮定

- $P, xP, x^2P, \dots, x^qP \in G$ から, ある c に対する $(c, e(P, P)^{1/(x+c)})$ を求めるのが困難
- 原型は私たちの2001年の放送型暗号の提案に基づく安全性仮定
 - $\left\{ \frac{1}{x+a_1}g, \dots, \frac{1}{x+a_n}g \right\}$ から $\frac{1}{x+a}g (a \notin \{a_1, \dots, a_n\})$ を求める問題A
 - $\{P, xP, \dots, x^qP\}$ から $(1/x)P$ や $x^{q+1}P$ を求める問題Bと同値であることを示す
 - 特に $\{P, xP\}$ から x^2P を求める問題はDHP問題と同値なので, 問題A, Bも困難と予想
- BBS署名の論文 in Section 3.2. The SDH assumption was recently used by Boneh and Boyen to construct short signatures without random oracles [8]. A closely related assumption was used by Mitsunari et al. [23] to construct a traitor-tracing system. The SDH assumption has similar properties to the

問題

- P, xP, yP から xyP を求める困難さと P, xP から x^2P を求める困難さは同程度であることを示せ