OpenID Connectを用いた スマートコントラクト代理実行における身元秘匿と本人同意の検証

◯平山洪[1] 掛井将平[1] 齋藤彰一[1]

1 名古屋工業大学

背景・問題

- 鍵管理・取引処理の委託
  - ブロックチェーンの利用には、秘密鍵の管理や取引操作が必要 → ユーザの負担を軽減する委託の枠組み
- 既存IDでの代理実行における関連技術：zkLogin
  - OpenID Connectアカウント（Google等）でブロックチェーンを利用可能にする手法
  - 認証（JWTの正当性・アドレス導出）は暗号学的に保証
- **問題**：委託事業者の不正や侵害による実行内容の差し替えを検知できない
  - 認証後に実行するトランザクションは委託サービス側が生成
  - 利用者が認証時に意図した内容と一致する保証がない

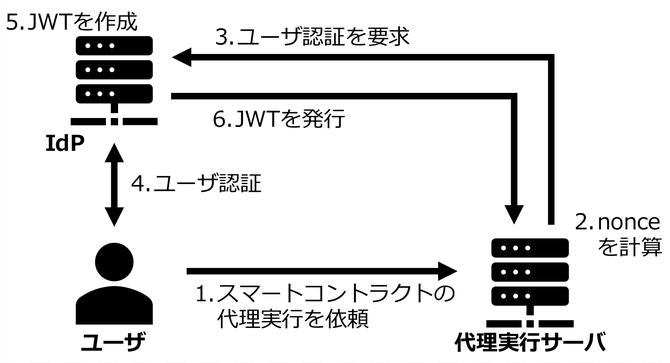
目的・課題

- **目的**：委託による代理実行において，同意内容と実行内容の一致を保証
  - 同意してない実行をスマートコントラクトの委託実行前に検知して拒否
  - 既存IDの利便性を保ちつつ委託事業者への信頼なしに一致を保証する
- **課題**：既存IDによる認証結果に同意内容を暗号学的に結び付け、オンチェーンで検証する
  - ユーザ自身の署名を同意の証拠に使わない（ユーザは専用の鍵を持たない）
  - 実行主体である委託サービスの正しさにも依存しない

提案：OpenID Connectとゼロ知識証明を用いたスマートコントラクト代理実行

①ユーザの同意の取得

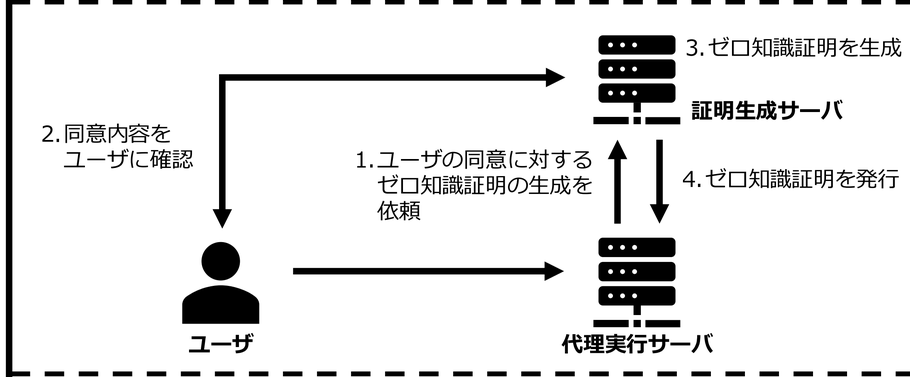
ユーザが同意した内容を
IdPの署名付き認証トークンに結びつけて
改ざんを検知可能にする

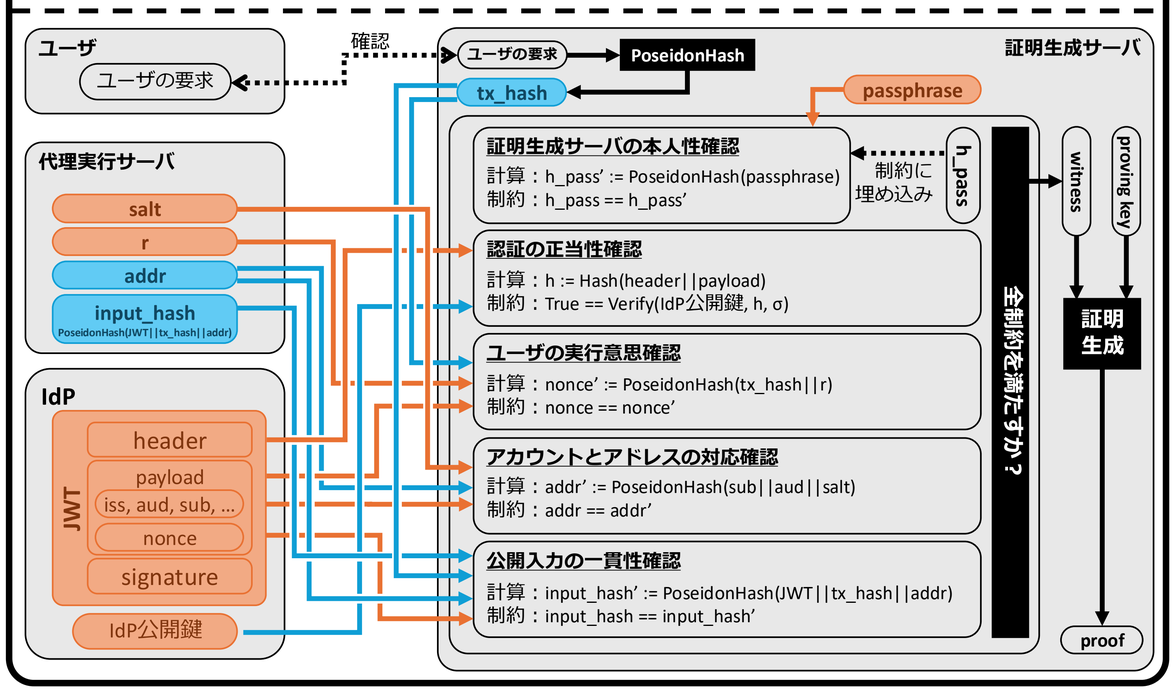




②ユーザの同意の証明の生成

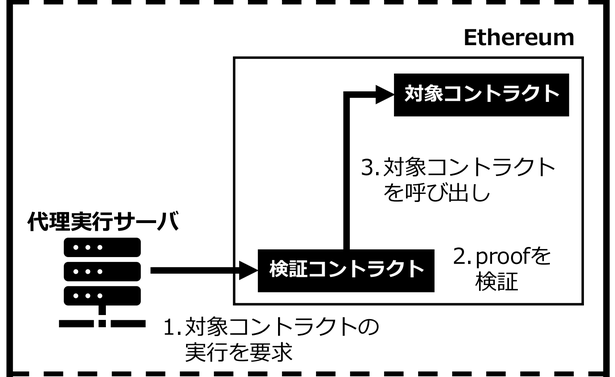
正規の認証トークンと同意した実行内容が
結びついていることを示すゼロ知識証明を生成する

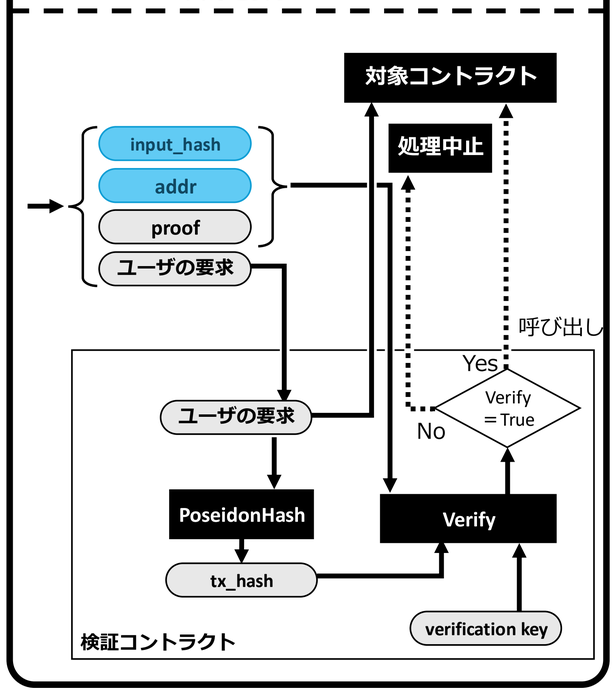




③ユーザの同意の検証

認証トークンを公開せずに
ゼロ知識証明から同意内容の検証をして
一致した時のみコントラクトを呼び出す





評価：ユーザ要求からコントラクト実行までの各フェーズに要する時間の評価

計測時間	平均	割合	
Nonce生成	0.178 ms	0.002%	①
JWT生成	0.807 ms	0.010%	
Witness生成	3,402.413 ms	43.267%	②
Proof生成	4,195.528 ms	53.353%	
Verify	186.514 ms	2.372%	③
Transaction実行	72.619 ms	0.923%	
Total	7,863.732 ms	100.000%	

測定結果：100回平均

合計時間はセットアップなしで7.8秒程度
その96%以上が**②ユーザの同意の証明の生成**の部分
②の計算量が高く、snarkjsでは最大CPU使用率890%
回路の制約数：1,048,974 = $2^{20}+397$
制約数を2^{20}以下に抑えればkey容量と計算コストが削減
今後、制約数削減による高速化や計算量の削減を目指す

測定環境

macBook Pro（macOS 26.0) Apple M5(10コア) 16GBメモリ

謝辞：本研究は JP25K21199 の助成を受けたものです。