Vol.2022-CSEC-99 No.16
Vol.2022-SPT-49 No.16
Vol.2022-EIP-98 No.16
2022/12/23

デジタル署名を活用したデジタルデータ権利管理システムと順序情報を保証するハッシュチェーン型集約署名の提案

金子真由子[1,*]　小栗秀暢[1]　小嶋陸大[1]　宇治橋善史[1]　片山佳則[1]
佐々木佑樹[1]　鈴木敦也[1]　鄭明燮[1]　中村裕[1]　芳賀翔太[1]　野村佳秀[1]

概要：近年，デジタルコンテンツに対して利用権などの権利管理情報を保証する手段として NFT が注目されている．しかし NFT を用いた権利管理はブロックチェーン・仮想通貨を用いることからオラクル問題や運用コスト等の課題が多く提起されている．本稿では，それらの課題をデジタル署名技術の活用によって解決する枠組みを提案する．デジタル署名型の権利管理の実現には，IdP やサービス事業者などと連携することで，本人情報とデジタルデータの真正性が定まることが必要である．それにより，例えばサービス間をまたいだ場合でもデータの一意性を保証することができ，サービスに紐づけられた権利との対応が明確となる．しかし，既存の NFT 技術では所有者の移転情報をブロックチェーンに記録することで，権利の移転や変更状況を明確化しており，デジタル署名技術だけでは代替が困難だった．そこで，我々はデジタルデータの権利管理に適した集約署名を提案する．我々が要件として定めたのは，順序が検証可能であること，署名人数にかかわらず集約署名の検証にかかる時間が短いことなどである．これらの要件を解決する方式としてハッシュチェーン型集約署名を考案した．本方式は，署名時に全署名者のメッセージハッシュを使い，署名集約時にはメッセージハッシュに対して署名することで，署名順序の保証と耐改ざん性を実現している．本稿では，提案した署名技術の有効性と，性能測定実験結果から，提案方式がサーバ台数に比例して TPS を向上できること，NFT よりも消費電力量が少ない上にスケールダウンも可能であること，署名人数にかかわらず集約署名の検証に掛かる時間が概ね一定であることを示し，実際の使用に耐えることを検証した．

1. はじめに

近年，インターネットの様々な領域において，従来の中央集権的なサービス提供型プラットフォームからの脱却を目指す，Web3 型のビジネスが注目されている[1]．その中でも，特に NFT を用いたデジタルコンテンツ市場は日本国内でも注目されており，例えば，自民党デジタル社会推進本部によると，NFT は Web3 時代の経済圏を拡大する起爆剤と考えられている[2].

しかし，NFT による所有権管理を行うサービスには，オラクル問題や運用コスト面の問題など複数の問題が生じることが知られている[3]．今後ますます発展が予想されるメタバース・ゲーム・VR 等のサービスにおいて，それぞれが保持するアイテムやデータを，その権利も含めて安全に他のサービスや実社会で活用できる枠組みが求められている．

そこで本稿では，メタバースやゲーム，サービス等の閉じられた世界で発行されたデータに対して，デジタル署名技術を用いて本人と紐づいた一意の ID を発行することでデジタルデータの権利管理を行うシステムを提案する．

また，その実現に必要なデジタル署名方式として，ブロックチェーンと同じくハッシュチェーンのつながりによって，利用者の取引プロセスを検証可能とするデジタル署名技術（ハッシュチェーン型集約署名）を提案する．

その際に，デジタル署名だけではカバーできない課題である，本人確認の方法や決済情報については，他のプラットフォームと連携して判断可能な構成とすることで，安全なデジタルデータの権利管理を実現する．

2. NFT による権利管理の課題

NFT とは，特定のデータや物理的な物体との紐づけ情報を，主に分散台帳であるブロックチェーンに記載することで，売買または授受が可能となったデータ単位である[4]．現在の市場では，デジタルコンテンツと紐づいた NFT に，著作権・所有権・利用権などの特定の権利（本稿では 「利用権など」と統一する）を付与して取引している．

しかし，メタバースやゲーム等の権利管理技術として NFT を用いた場合，以下の(A)～(F)の課題がある．

2.1 機能に関する課題

(A) オラクル問題

NFT には，ブロックチェーン外部に存在するコンテンツの信頼性を保証できない「オラクル問題」が存在する[5]．これによって例えば，当該コンテンツの権利を有さない者による NFT 発行ができることや，ブロックチェーン外の NFT メタデータやコンテンツ本体の改ざんができることが問題となる[6].

(B) 二重譲渡の発生

運用コストの都合により，コンテンツをオフチェーン（保有者のアドレスと，メタデータの URL が記載されたインデックスデータのみをブロックチェーン上で記録する方式）で管理する NFT サービスも多い．このとき，オラクル問題を応用して 1 つのコンテンツを同時に複数の NFT マーケットで販売することができてしまう[7].

(C) 投機的要因による価格変動の発生（「ガス代」問題）

NFT の取引において,取引結果をブロックチェーン上の台帳に記録する時にガス代が発生する．ブロックチェーン

1 富士通株式会社
Fujitsu Limited
* k.mayuko@fujitsu.com, fj-taas_pj-all@dl.jp.fujitsu.com

Vol.2022-CSEC-99 No.16
Vol.2022-SPT-49 No.16
Vol.2022-EIP-98 No.16
2022/12/23

上の取引が増加し，トランザクション数が多くなるとネットワークへの負担が大きくなり，取引承認のスピードが落ちてしまう.ガス代を高く設定することで，取引承認のためのマイニングを優先的に受けられることから，ガス代が高騰し，少額決済においても高額なガス代が必要となるケースが増加している[8].

(D) 返金等の対応が困難

NFT サービスに利用されるブロックチェーンには NFT トークン以外にも暗号資産のトランザクションも多く含まれており，それらすべてを含めた１つの処理の真正性を保証している．そのため，ブロックチェーンに登録されたトークンを修正しようとすると他の処理にもその影響が及ぶことから，ロールバックや返金処理は困難である[9]. また，暗号資産取引には責任主体や監査摘発主体が不在のため，資金洗浄に利用される可能性が指摘されている．

2.2 性能に関する課題

(E) トランザクション性能の問題

ブロックチェーンにおける合意メカニズム（PoW，Proof of Stakes（PoS）等）は，ネットワーク負荷状況が高い場合に確認遅延や確認失敗が起こりうる[10].

(F) 消費電力量が高い（サステナビリティの課題）

ブロックチェーンの基本原理である Proof of Work（PoW）方式はトランザクション検証時のマイニングにおいて大量の計算処理が必要となることから，計算機による消費電力量が増大している[11].

3. デジタル署名を用いた権利管理システム

本稿では，上記 NFT の課題を解決する枠組みとして，デジタル署名技術を活用するシステムを提案する．これによって，あるユーザが，ある一意のデジタルデータを，あるサービス(ゲーム・メタバース等)の中で利用可能であることを証明する. サービスの中での一意性を担保することで，そのデジタルデータに紐づけられたサービス内での利用権などの権利との紐づけが可能となる．

3.1 機能要件・性能要件のまとめ

2.1 節で挙げた課題を集約すると以下の通りである. 本技術はこれらの課題を解決する必要がある．

表 3.1-1 機能要件・性能要件

機能要件	(A)オラクル問題の防止
	(B)二重譲渡の防止
	(C)投機的要因による価格変動の防止
	(D)返金対応等が容易であること
性能要件	(E) 既存の NFT と同等以上のトランザクション性能
	(F) 既存の NFT よりも消費電力量を抑えられること

3.2 サービスの組み合わせによる実現

前項の要件を満たすための，デジタル署名システムと他のシステムとの関係性を以下に示す.

本システムの特徴は次の通り．

- デジタル署名型権利管理システムは，1.デジタル署名発行システム，2. ゲームやメタバース等のサービス事業者，3. ID/属性認証システム が連携し，従来の NFT とは異なる, 明確な管理者がいる集中管理型のシステム構成となる．
- ID/属性認証システム，ゲーム/メタバース基盤，デジタル署名発行システム, およびそれらを利用する開発者・利用者（ゲーム会社・ゲームユーザ等）で構成される．

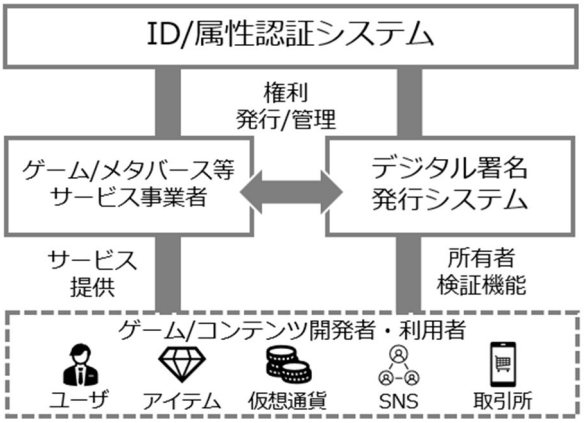

図 3.2-1 デジタル署名型権利管理システムの関係図

各要素の役割は以下の通り．

- 開発者がメタバース（ゲーム）を開発. メタバース基盤が提供する API を用いて，開発者がメタバース基盤上で動作するメタバース（ゲーム）を作成し，公開・運営を行う．
- ユーザはメタバース基盤に自身のアカウントを登録して基盤上の ID を取得する．
  - これによりメタバース基盤はユーザ識別が可能になる．
- メタバース基盤または開発者が，ユーザ ID・デジタルコンテンツ ID・メタバース内のルールを記録する DB を管理する．
- デジタル署名発行システムは, コンテンツの取引が発生した際に，規定された順序で組織署名を発行する．
  - デジタル署名発行システム・メタバース基盤・管理 DB の連携により，署名情報と生成・譲渡の情報を用いて取引の真正性を検証することが可能になる．
  - コンテンツデータ内に署名を埋め込むことで，NFT のようなコンテンツとトークンの乖離を防ぐ.
- ID/属性認証システムは決済に関する情報を記録.
  - デジタル署名発行システムを連携させることで，検証処理で不正な取引が発覚した場合に，コン

Vol.2022-CSEC-99 No.16
Vol.2022-SPT-49 No.16
Vol.2022-EIP-98 No.16
2022/12/23

テンツ及び所持ユーザの決済記録の問い合わせによる返金処理やロールバックが可能となる．

3.3 本システムによって解決する課題

(A) オラクル問題

本提案ではユーザを認証したうえで，ゲーム内コンテンツの生成や利用者間での譲渡等を一通り追跡可能であるため，複数のゲームにわたって常に正当な利用者が特定コンテンツを所持していることを保証できる．また出自や譲渡の経路が不明なコンテンツの生成・利用を禁止するルールを設定しておくことで，不正なデジタルコンテンツの発行を防止できる．

(B) 二重譲渡の問題

オラクル問題と同様に，ユーザ認証を通してあらゆるコンテンツの取引を把握し，不正を検知することができる．

またブロックチェーンと比べて省電力で同等のトランザクション量を処理することができる（5 章にて詳述）．また要求されるトランザクション量が少なければサーバ台数をスケールダウンすることもでき，省電力化も可能である．

(C) 投機的要因による価格変動の発生

本提案は中央集権型システムであり，取引における運用費やガス代を運営側でマネジメントできるため，安定したコストでサービスを提供することが可能となる．

(D) 返金対応等の困難性

本提案では集約署名を用いて取引ごとの真正性を検証するため，ある取引について取消しを行うことが NFT と比べて容易である．またメタバース基盤が責任主体として存在するため，定められたガバナンスに基づいて不正や犯罪に対処することができる．

(E) トランザクション性能の課題

本提案の TPS はサーバ性能に比例して向上する．そのため，サービス要件に応じて十分な性能のサーバを用意することで，NFT を代替することができる．

(F) 消費電力量の課題

ブロックチェーンと比べて省電力で同等のトランザクション量を処理することができる．また要求されるトランザクション量が少なければサーバ台数をスケールダウンすることもでき，省電力化も可能である．

3.4 デジタル署名を用いた権利管理システムの課題

本章において，デジタル署名型権利管理システムは，2 章の(A)-(F)の課題を解決することを述べた．

しかし，次章で述べるが，従来型のデジタル署名のみを用いると，誰がどのような順番で取引を行ったかが扱えない．また，全員の署名の検証を行なわなければならず，高速な処理ができない．そのため，署名の順序性と集約性が必要である．順序の検証と取引履歴情報の保持を実現させるための署名技術を 4 章で検討，提案する．

4. 順序の検証と取引履歴情報の保持を実現する署名 ~ ハッシュチェーン型集約署名

4.1 要件

デジタル署名による権利管理処理を行う際に，その取引の順序情報を保持するためには，順序性と集約性を持つデジタル署名方式が必要である．

- 順序性とは，コンテンツ生成や譲渡に関わった複数当事者の署名順序を証明できることである．これにより，当事者全員の合意された取引プロセスであることの真正性やコンテンツ来歴の保証を可能にする．
- 集約性は，この複数署名を利用した譲渡の真正性検証を単一処理に集約できることである．これにより高速で簡潔な検証処理を実現する．

また，集約署名技術には，Rogue-key attack 攻撃があることが知られている[12][13][14]．このため，デジタル署名による権利管理を実現する集約署名には，この攻撃に対する耐性が必要である．

したがって，次の３つの機能要件を満たす必要がある．

機能要件：
(1)単一のデジタルデータに対して付与された複数の署名を，単一の処理で検証可能．
(2) 複数の署名付与に際して，各署名者による署名付与の順序を検証可能．
(3) 検証者は検証時に Rogue-key Attack への耐性があるような集約検証鍵が利用可能．

さらに本方式では，今後の仕様拡充により署名数が増大しても署名・検証処理時間が増大しないことが望ましい．例えば，コンテンツ移転時にペアレンタルコントロールのため親の署名も追加する，等の拡充が考えられる．これらを整理すると，以下の性能要件が必要になる．

性能要件：
(4) 検証者が署名数に依存せず，定数時間で検証を完了する．
(5) 署名者が署名数に依存せず，定数時間で署名を完了する．

本方式で利用する集約署名技術は，以上の(1)～(5)の要件を満たしていることが望ましい．

4.2 従来技術との比較

従来技術として以下に挙げられる方式があるが，　(1)～(5)の要件すべてを満たす方式は存在していない．

IAS: Interactive Aggregate Signature [13] [15]

一般に Multi-Signature (MS) は同一メッセージに対する署名のみサポートする．複数のメッセージに対する署名をサポートするような MS の拡張方式を IAS と呼ぶ．これは[15]で提案され，[13]では安全性を向上させる Secure IAS が提案された．

MS では単一のメッセージに対して署名するが，IAS ではメッセージの集合Sを構成し，各署名者はSに対して高々1 回署名を付与している．

Vol.2022-CSEC-99 No.16
Vol.2022-SPT-49 No.16
Vol.2022-EIP-98 No.16
2022/12/23

よって IAS は上記(1)(5)の要件を満たすが，(2)集合内の順序情報は検証不可，(3)集約検証鍵を用いた検証は不可，(4)検証には全ての検証鍵が必要なため，他の要件を満たさない．

OAS: Optimized Aggregate Signature [16]

OAS [16] は MS と Aggregate Signature (AS) を一般化させた署名処理である．複数署名者の間で Default message Mを決定し，もし署名者がMに署名を付与する場合は MS の集約アルゴリズム，それ以外のメッセージに対して署名する場合は AS の集約アルゴリズムを用いて集約処理を行う．すなわち，全ての署名者がMに対して署名するならばベストパフォーマンスを発揮するが，全ての署名者が異なるメッセージに対して署名する場合，これは AS と等しくなる．よって OAS は(1)(5)，条件によっては(4)は満たすが，(2)各署名の順序情報は検証不可，(3)集約検証鍵を利用はするが Rouge-key Attack への対策は考慮されていないため, 他の要件を満たさない．

AMSP: Aggregate Multi-Signature Protocol [14]

AMSP[14]は各メッセージに対する集約署名を作成し, その集約署名をさらに集約することで集約署名を計算する．最終的に出力される集約署名は集約検証鍵を用いて検証可能であり，各検証鍵には PoPs: Proof of Possessions が付与されているため Rogue-key Attack への耐性がある．

よって(1)(3)(4)を満たすが，(2)各署名の順序情報は検証不可，(5) 各署名者はn（署名者数）回の署名付与が必要となるため，他の要件を満たさない．

SAS: Sequential Aggregate Signature [17]

SAS は AS の一種で，署名の作成に対して順序性が導入されている．

よって(1)(2)(5)を満たすが，(3)集約検証鍵を用いた検証は不可，(4)検証には全ての検証鍵を必要とするため，他の要件を満たさない．

KAIAS: Key Aggregatable IAS [18]

KAIAS は Message Aggregation という仕組みを導入することで，集約検証鍵を用いて検証可能な IAS である．また集約検証鍵は[13]と同様に，Plain Publickey model に基づく集約検証鍵が利用可能なため, Rogue-key Attack への耐性がある．

よって(1)(3)(4)(5)を満たすが，(2)各署名の順序情報は検証不可である．

4.3 提案手法

ハッシュチェーン型集約署名は従来の集約署名技術と異なり，署名時に全署名者のメッセージハッシュを使い，さらに署名集約時にはメッセージハッシュに対して署名している点がポイントとなる．

以下では署名アルゴリズムの詳細を説明する．まず前提となる KAIAS の方式を 4.3.1 節で説明したうえで，提案手法を 4.3.2 節で説明する．

4.3.1 KAIAS の構成

3-round かつ鍵集約可能な IAS スキームである$\mathrm{KAIAS} = (\mathrm{Pg}, \mathrm{Kg}, \mathrm{KAg}, \mathrm{Sign}, \mathrm{SAg}, \mathrm{MHAg}, \mathrm{KVf}, \mathrm{AVf})$

及びハッシュ関数 H_0, H_1, H_2 を以下で定義する．このスキームは署名者n人，検証者 1 人，「集約者」1 人の間で実行される．集約者は，署名の集約と検証鍵の集約を代行する署名者側のエンティティであり，署名者の 1 人がこれを代行してもよい．また集約は署名者間で共有されるパブリックな情報のみを元に計算されるため，少なくとも 1 人の semi-honest な署名者が存在すれば，集約計算の正当性確認が可能である．

位数pの整数剰余環を$\mathbb{Z}/\mathrm{p}\mathbb{Z}$で表記する．各署名者にはインデックスが与えられており，i-番目の署名者を署名者- iで表記する．

Hash Function

それぞれハッシュ関数H_0, H_1, H_2を

$\mathrm{H_i}: \{0,1\}^* \to \mathbb{Z}/\mathrm{p}\mathbb{Z}¥\{0\}\ (\mathrm{i} \in \{0,1,2\})$で定義する（値域は非零とする）．

Parameter Generation. $\boldsymbol{KAIAS.Pg(1^\kappa)}$

このアルゴリズムでは各アルゴリズムが共通で利用するパラメータ$params$を出力する．

セキュリティパラメータκを入力とする. κ-bit のpを位数として持つような巡回群を$\boldsymbol{G}$とおき，その生成元をgとおく．$params = (\boldsymbol{G}, p, g)$を Group Parameter として出力する．

Key Generation. $\boldsymbol{KAIAS.Kg(params)}$

このアルゴリズムでは各署名者の鍵ペアを生成する．

署名者-iは一様ランダムな署名鍵$x_i \leftarrow \mathbb{Z}/\mathrm{p}\mathbb{Z}$ を取得し，検証鍵$X_i \leftarrow g^{x_i}$を計算する．鍵ペア$(sk = x_i, pk = X_i)$を出力する．$L_{pk} \leftarrow \{X_1, \cdots, X_n\}$を検証鍵リストとおく．

Key Aggregation. $\boldsymbol{KAIAS.KAg(L_{pk})}$

このアルゴリズムでは検証鍵リストを受け取り，集約検証鍵を出力する．

各署名者-iは$a_i \leftarrow H_1(X_i, L_{pk})$を計算し, a_iを集約者へ送信する．集約者は集約検証鍵$\tilde{X} \leftarrow \prod_{i=1}^{n} X_i^{a_i}$を計算し，$\tilde{X}$を出力する．

Signing. $\boldsymbol{KAIAS.Sign(params, L_{pk}, sk_i, m_i)}$

このアルゴリズムでは以下の 3-Round プロトコルを通じて署名σ_iを出力する．

1. 署名者-iは一様な乱数$r_i \leftarrow \mathbb{Z}/\mathrm{p}\mathbb{Z}$を取得し，$R_i \leftarrow g^{r_i}, t_i \leftarrow H_2(R_i)$を計算する．署名者-$i$は$t_i$をコミットメントとして集約者に送信する．集約者は$L_T \leftarrow \{t_1, \cdots, t_n\}$を各署名者に送信する．
2. 署名者-iはR_iを集約者に送信する．集約者は$L_R \leftarrow \{R_1, \cdots, R_n\}$を各署名者に送信する．全ての署名者-$i$は$j \in \{1, \cdots, i-1, i+1, \cdots, n\}$に対し，$t_j \leftarrow H_2(R_j)$が等しいか検証する．検証に失敗した場合，プロトコルを中止する．
3. 集約者は$\tilde{X} \leftarrow KAg(L_{pk})$を計算し，$(\tilde{X}, \{a_1, \cdots, a_n\})$（ただし$a_i \leftarrow H_1(X_i, L_{pk})$）を計算する．

その後，署名者-iは以下の手順で自身の所持するメッセー

Vol.2022-CSEC-99 No.16
Vol.2022-SPT-49 No.16
Vol.2022-EIP-98 No.16
2022/12/23

ジm_iに対する署名σ_iを計算する．

$$\overline{R} \leftarrow \prod_{i=1}^{n} R_i$$

$$c_i \leftarrow H_0(\overline{R}, \widetilde{X}, m_i)$$

$$a_i \leftarrow H_1(X_i, L_{pk})$$

$$s_i \leftarrow r_i + c_i \cdot a_i \cdot x_i \bmod p$$

署名者は$\sigma_i = (s_i, \overline{R})$を計算し，$(c_i, \sigma_i)$を集約者へ送信する

Signature Aggregation.

$KAIAS.SAg(params, \{(c_1, \sigma_1), \cdots, (c_n, \sigma_n)\})$

このアルゴリズムでは集約者が以下の手順で集約署名$\tilde{\sigma}$を計算し出力する．

$$\tilde{s} \leftarrow \sum_{i=1}^{n} s_i \cdot d_i,\ \widetilde{R} \leftarrow \prod_{i=1}^{n} R_i^{d_i}$$

ここで任意のiについて，d_iは以下で定義される．

$$d_i \leftarrow \prod_{j \in \{1,\cdots,n\}-i} c_j = c_1 \cdots c_{i-1} \cdot c_{i+1} \cdots c_n$$

集約者は$\tilde{\sigma} = (\tilde{s}, \tilde{R}, \overline{R})$を出力する．

Message Hash Aggregation.$KAIAS.MHAg(\{m_1, \cdots, m_n\}, \overline{R}, \tilde{X})$

このアルゴリズムは検証者が以下のような集約メッセージハッシュ$\overline{c}$を計算し出力する．

$$\overline{c} \leftarrow \prod_{i=1}^{n} c_i$$

ただし任意のiについて，$c_i \leftarrow H_0(\overline{R}, \tilde{X}, m_i)$.

Key Verification. $KAIAS.KVf(L_{pk}, \tilde{X})$

このアルゴリズムは検証者が集約検証鍵の正当性を確認するために用いる．

$\tilde{X} \leftarrow KAIAS.KAg(L_{pk})$ならば 1 を出力し，そうでなければ 0 を出力する．

Aggregated Signature Verification.

$KAIAS.AVf(\{m_1, \cdots, m_n\}, \tilde{\sigma}, \tilde{X})$

このアルゴリズムは検証者が以下の手順に従い，集約署名を検証するために用いる．

1. $\overline{c} \leftarrow KAIAS.MHAg(\{m_1, \cdots, m_n\}, \overline{R}, \tilde{X})$を計算する．
2. $g^{\tilde{s}} = \tilde{R} \cdot \tilde{X}^{\overline{c}}$が成り立つならば 1 を出力，そうでなければ 0 を出力する．

Correctness は以下で確認される．

$$\begin{aligned} g^{\tilde{s}} &= g^{s_1 d_1 + \cdots + s_n d_n} \\ &= g^{r_1 d_1 + \cdots + r_n d_n + c_1 d_1 a_1 x_1 + \cdots + c_n d_n a_n x_n} \\ &= R_1^{d_1} \cdots R_n^{d_n} \cdot (X_1^{a_1} \cdots X_n^{a_n})^{\overline{c}} \\ &= \widetilde{R} \cdot \widetilde{X}^{\overline{c}} \end{aligned}$$

$$\begin{aligned} g^{\tilde{s}} &= g^{s_1 d_1 + \cdots + s_n d_n} \\ &= g^{r_1 d_1 + \cdots + r_n d_n + c_1 d_1 a_1 x_1 + \cdots + c_n d_n a_n x_n} \\ &= R_1^{d_1} \cdots R_n^{d_n} \cdot (X_1^{a_1} \cdots X_n^{a_n})^{\overline{c}} \\ &= \widetilde{R} \cdot \widetilde{X}^{\overline{c}} \end{aligned}$$

4.3.2 ハッシュチェーンとの組み合わせ

次にハッシュチェーンを組み合わせることで署名の順序性を導入するようなスキームとして$KAIAS^{+} = KAIAS + (HCSign, HCVf, Vf)$を以下で定義する．

本スキームでは 4.3.1 節との違いとして，集約者が署名処理に参加することを想定する．

すなわち，集約者が署名者-$(n+1)$として参加し，$(x_{n+1}, X_{n+1}) \leftarrow KAIAS.Kg(params)$で鍵ペアを生成し，検証鍵リスト$L_{pk} = \{X_1, \cdots, X_n, X_{n+1}\}$から$KAIAS.KAg(L_{pk})$を作成する．

これは 4.4.1 節のnを$n+1$に置き換えることで自然な読み替えが可能である．

Hash Chain Signing. $HCSign(params, L_{pk}, sk_i, m_i)$

このアルゴリズムはハッシュチェーンリストL_{hc}を出力する．

まず $KAIAS.Sign(params, L_{pk}, sk_i, m_i)$（ただし $i \in \{1, \cdots, n+1\}$）の 3-Round プロトコルまでは共通の処理を行う．

次に$i = 1$の場合，署名者-1は以下を計算する．

$$\overline{R} \leftarrow \prod_{i=1}^{n} R_i$$

$$h_1 \leftarrow H_0(\overline{R}, \widetilde{X}, m_1)$$

$$c_1 \leftarrow h_1$$

$$a_1 \leftarrow H_1(X_1, L_{pk})$$

$$s_1 \leftarrow r_1 + c_1 \cdot a_1 \cdot x_1 \bmod p$$

署名者-1は$\sigma_i = (s_1, R_1)$として(c_1, σ_1)を集約者へ送信する．

$i = k\ (2 \le k < n+1)$の場合，署名者-kは以下を計算する．

$$\overline{R} \leftarrow \prod_{i=1}^{n} R_i$$

$$h_k \leftarrow H_0(\overline{R}, \widetilde{X}, m_k)$$

$$c_k \leftarrow H_1(c_{k-1}, h_k)$$

$$a_k \leftarrow H_1(X_k, L_{pk})$$

$$s_k \leftarrow r_k + c_k \cdot a_k \cdot x_k \bmod p$$

署名者-kは$\sigma_k = (s_k, \overline{R})$として$(c_k, \sigma_k)$を集約者へ送信する．

最後に$i = n+1$の場合，署名者-$n+1$，すなわち集約者はメッセージではなくc_nに対する署名として以下を計算する．

$$\overline{R} \leftarrow \prod_{i=1}^{n} R_i$$

$$h_{n+1} \leftarrow H_0(\overline{R}, \widetilde{X}, c_n)$$

$$c_{n+1} \leftarrow H_1(c_n, h_{n+1})$$

$$a_{n+1} \leftarrow H_1(X_{n+1}, L_{pk})$$

$$s_{n+1} \leftarrow r_{n+1} + c_{n+1} \cdot a_{n+1} \cdot x_{n+1} \bmod p$$

集約者は$\sigma_{n+1} = (s_{n+1}, \overline{R})$として$(c_{n+1}, \sigma_{n+1})$を得る．

Hash Chain Verify. $HCVf(\{m_1, \cdots, m_n, c_n\}, \tilde{\sigma}, \tilde{X})$

このアルゴリズムでは，以下の手順でハッシュチェーンを検証する．

Vol.2022-CSEC-99 No.16
Vol.2022-SPT-49 No.16
Vol.2022-EIP-98 No.16
2022/12/23

$h'_1 \leftarrow H_0(\overline{R}, \widetilde{X}, m_1)$
$c'_1 \leftarrow h'_1$
$h'_2 \leftarrow H_0(\overline{R}, \widetilde{X}, m_2)$
$c'_2 \leftarrow H_1(c'_1, h'_2)$
$\vdots$
$h'_n \leftarrow H_0(\overline{R}, \widetilde{X}, m_n)$
$c'_n \leftarrow H_1(c'_{n-1}, h'_n)$

最終的に$c_n = c'_n$が成立するならば 1 を出力，そうでなければ 0 を出力する．

Verify. $Vf(\{m_1, \cdots, m_n, c_n\}, \tilde{\sigma}, \tilde{X})$

このアルゴリズムは$KAIAS.AVf(\{m_1, \cdots, m_n, c_n\}, \tilde{\sigma}, \tilde{X}) = 1$かつ$HCVf(\{m_1, \cdots, m_n, c_n\}, \tilde{\sigma}, \tilde{X}) = 1$の場合 1 を出力し，そうでなければ 0 を出力する．

本方式に関する（インフォーマルな）説明を付随する．本検証アルゴリズムについて，$KAIAS.AVf$ は非改ざん性と署名者の本人性，$KAIAS.HCVf$では署名順序の正当性をそれぞれ検証する．

しかし$KAIAS.HCVf$ のみの検証では，以下の手順により任意のメッセージに対する改ざんが可能である．

- 攻撃者は任意のiについて，メッセージm_iを$\dot{m}_i$に改ざんする．
- 攻撃者は$\dot{h}_i \leftarrow H_0(\overline{R}, \tilde{X}, \dot{m}_i)$, $\dot{c}_i \leftarrow H_1(c_{k-1}, \dot{h}_i)$（ただし$i = 1$ならば$\dot{c}_i \leftarrow \dot{h}_i$）を計算する．
- 攻撃者は以降$k > i$について$\dot{c}_k \leftarrow H_1(\dot{c}_{k-1}, h_k)$を計算する．
- 攻撃者はc_nを$\dot{c}_n$に改ざんすることで，$KAIAS.HCVf(\{m_1, \cdots, \dot{m}_i, \cdots, m_n, \dot{c}_n\}, \tilde{\sigma}, \tilde{X}) = 1$を満たすような$\{m_1, \cdots, \dot{m}_i, \cdots, m_n, \dot{c}_n\}$を構成可能．

そこで本方式ではc_nを$KAIAS.AVf$の署名対象として含めることにより，c_nに非改ざん性を付与している．

これにより$KAIAS.AVf$と$KAIAS.HCVf$の検証の両方に成功する場合に限り，署名対象のデータは非改ざん性，署名者の本人性，署名順序の正当性の 3 点を満たす．よって 4.1 で述べた要件(1)(3)(4)(5)に加え，署名順序の正当性(2)を満たすような方式が構成された．

5. 性能調査

本章では，デジタル署名による権利管理のコア技術となるハッシュチェーン型集約署名のトランザクション性能について調査し，既存の署名技術・ブロックチェーン技術のトランザクション性能との比較を行った．

5.1 ブロックチェーンのトランザクション性能調査

比較対象とするブロックチェーンは，仮想通貨の分析ツールである DappRader を参照し，取引件数の多い NFT マーケットプレイスに使用されているものを選択した．今回は，チャートサイト（BLOCKCHAIR, Solanabeach, Polygonscan, bscscan.com， flowscan.org， tzstats.com， ethtps.info， wax.blocks.io）で確認された TPS 値をトランザクション性能の実測値とした．

調査結果となる各 TPS 値，もしくは 1 時間の平均取引件数から算出した値とコンセンサス・アルゴリズム形式について表 5.1-1 にまとめる．結果から，現在高速と言われている方式を採用した BC が現状社会的に処理しているトランザクション量は秒間 2000-3000 件程であると分かる．

表 5.1-1 BC 基盤毎の実測 TPS

BC	形式	実測 [TPS]
SOLANA	PoS, PoH	3000
WAX	DPoS	200-2500
POLYGON	PoS	200-400 (1 トランザクションを 250 バイトと仮定した場合)
BNB CHAIN	PoSA(DPoS と PoA)	30-50
FLOW BLOCKCHAIN	PoS	22 (daily transaction が最も多かった日の数値を基に計算 2022/6/29 16:14)
ETHEREUM	PoW	12-20
RONIN	PoA	2-15
TEZOS	LPoS	00.89 (24 時間のトランザクション量から計算 tzStata 2022/6/29 16:07)

5.2 提案手法の性能評価

3 章および 4 章で述べた性能要件を満たしていることを本章で検証する．すなわち，

- トランザクション性能が NFT と同等以上の性能を達成可能であること（性能要件：3 章）
- 消費電力量が NFT と同等以下に抑制することが可能であること（性能要件：3 章）
- 署名/検証にかかる処理時間が署名数に非依存で定数時間で完了すること（集約署名性能要件：4 章）

表 5.2-1 性能試験で使用したサーバ環境

サーバ	AWS（Amazon Web Service）EC2
プロセッサ	第 1 または第 2 世代 Intel Xeon Platinum 8000
クロック周波数	最大 3.6 GHz
OS	Amazon Linux

表 5.2-2 使用したインスタンスの性能(vCPU 数，メモリ)

モデル	vCPU 数	メモリ (GB)
c5.large	2	4
c5.xlarge	4	8
c5.2xlarge	8	16
c5.4xlarge	16	32
c5.9xlarge	36	72
c5.18xlarge	72	144
c5.19xlarge	96	192

試験したプログラム：集約署名のコア部分（C 言語）を呼び出して表 5.3-1 の一連の動作を上から順に行う集約署名プログラム（java）を shell スクリプトでバックグラウンドによる実行を行った．

集約人数：3 名（サービス事業者，コンテンツの所有者，コ

Vol.2022-CSEC-99 No.16
Vol.2022-SPT-49 No.16
Vol.2022-EIP-98 No.16
2022/12/23

ンテンツの販売先の 3 者で同意して署名する例を想定）

試験方法：ハッシュチェーン型集約署名の一連の動作（鍵ペア生成～ハッシュチェーン検証）一式を実行するのに要した実行時間を計測．一連の動作を多数回繰り返し行い，これを順次（2 秒毎に）追加して同時に処理可能な数（多重度）を増やしていき，実行時間と同時実行数から単位時間（秒）あたりに実行できる数（TPS）を算出した．なお，繰り返し数と多重度はインスタンスの性能をあげるにつれて適宜増やしていった．

5.3 測定結果

5.3.1 トランザクション性能に関する測定実験

結果 1-1：多重度（1～10 まで）と TPS

試験結果グラフ[図 5.3-1]から以下のことが分かる．

- インスタンスの性能（vCPU 数）が上がると TPS が上がる
- 多重度（同時実行数）を増やすと TPS が上がるが，vCPU 数（PC では CPU のコア数）と同じくらい（グラフ上の〇印）で伸びが止まっている（PC は図 5.3-3 での比較のため掲載）

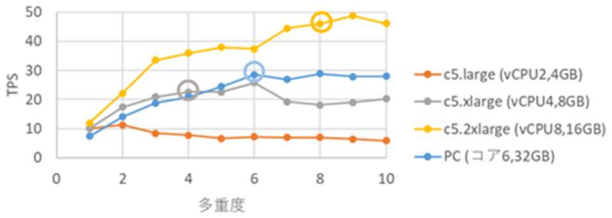

図 5.3-1　多重度（1～10 まで）と TPS

結果 1-2：多重度（1～150 まで）と TPS

（測定は TPS の伸びが止まるくらいまで行っているので，性能によって多重度の測定の上限が異なっている．）

試験結果グラフ[図 5.3-2]から以下のことが分かる．

- 結果 1-1 と同じ傾向だが，c5.24xlarge（vCPU96,192GB）では vCPU 数以上の多重度でも TPS が伸び続けている．140 多重付近まで伸び，600TPS を超えている．

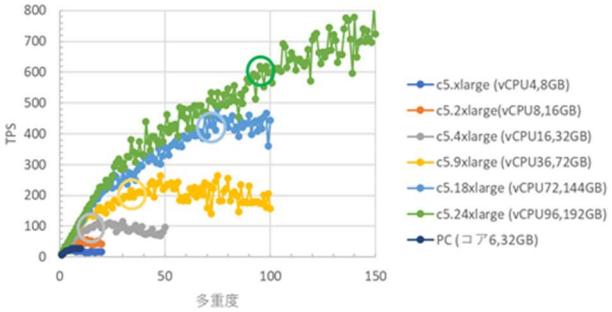

図 5.3-2　多重度（1～150 まで）と TPS

結果 1-3：性能（vCPU 数）と TPS

試験結果グラフ[図 5.3-3]から以下のことが分かる．

- インスタンスの性能（vCPU 数）が上がると直線状に TPS が上がっている．最大で 600TPS まで確認した．

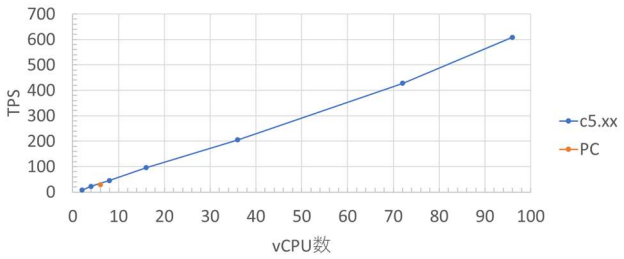

図 5.3-3　性能（vCPU 数）と TPS

5.3.2 署名人数と処理時間

結果 2：集約人数と処理時間

試験結果表[表 5.3-1]とグラフ[図 5.3-4,5]から以下が分かる．

- 集約署名の各工程は集約署名検証（⑥）を除き，集約人数が増えるとかかる時間も増える
  （測定環境：AWS EC2 c5.2xlarge，測定回数：1000 人，100 人→10 回，10～2 人→500 回の平均値）

表 5.3-1　集約人数と各工程にかかる時間（人, msec）

人数	2	10	100	1000
①鍵ペア生成	15.84	78.71	778.79	7769.82
②個人署名作成	10.00	47.43	458.03	4530.28
③個人署名検証	17.97	89.15	883.65	8826.21
④集約公開鍵作成	9.02	12.37	45.42	375.59
⑤集約署名作成	8.73	73.73	806.32	8085.56
⑥集約署名検証	8.46	8.50	8.57	8.56
⑦ハッシュチェーン検証	8.55	16.01	41.47	180.98
合計	78.57	325.90	3022.23	29777.00

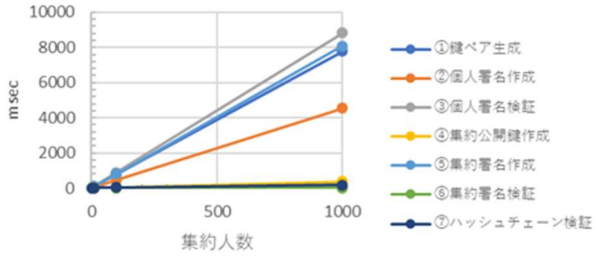

図 5.3-4　集約人数と計算時間（2, 10, 100, 1000 人）

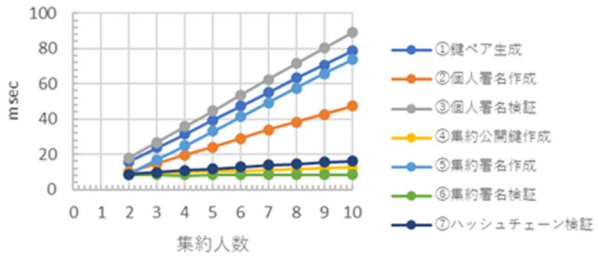

図 5.3-5　集約人数と計算時間（2～10 人）

5.4 測定実験結果に対する考察

5.4.1 トランザクション性能

vCPU の数に比例して TPS の数もリニアに大きくなっているため，サーバ性能およびサーバ数で処理可能なトランザクション量を決定することが可能である．そのため，新たなサービスを提供する場合，サービス要件として必要な

トランザクション量が把握できればシステム規模の見積もりが可能となる．また，ブロックチェーンシステムと異なり，並列環境の開発が容易である点もメリットである．

従来 NFT では実測値が最大 3000TPS 程度であり，我々の提案方式ではサーバを 5 台程度並列化すれば同等の規模のトランザクション量に対応可能であると考えられる．

5.4.2 消費電力量

NFT のマイニングにおけるコンセンサス・アルゴリズムの PoW では，1 回のトランザクションにおける電力消費量はイーサリアムが 134.38kWh，ビットコインが 1477.27kWh と大きい．我々の提案方式では 1 台のサーバマシンで数百 TPS の処理能力を達成でき，一台のサーバマシンを数百 Wh と仮定すると[19]，消費電力削減効果が得られる．

5.4.3 署名/検証にかかる時間の署名人数への依存性

提案方式の署名処理時間では集約署名の処理分，処理時間がかかっているが，検証処理時間は N 人分の検証を 1 回の検証で実現しているため，処理時間の短縮化になっている．

6. まとめ

本稿では，社会的に利用が進んでいる NFT 技術について，技術的な課題を整理し，その実用上の課題を解決するデジタル署名型の権利管理システムを提案した．

主たる構成として，デジタル署名発行システムと，ID/属性発行システムが，コンテンツプラットフォームやサービス事業者と信頼できる構成で連携しているとき，デジタル署名によって，利用ユーザ・デジタルデータ・サービス(サービス内の利用権など) を暗号的に結び付けることで，改ざんや不正利用を防ぐ権利管理システムが実現できることを示した．

しかし，既存の署名技術では，このシステムの実現に必要な要件として，利用ユーザの処理順序を保証することができないため，NFT の持つユーザ間の取引履歴を検証できないことが判明した．そこで我々は要件を満たすハッシュチェーン型集約署名を考案した．本署名方式は署名時に全署名者のメッセージハッシュを使ってハッシュ化を行い，署名集約時にはメッセージハッシュに対して署名することで，署名順序の保証と耐改ざん性を実現している．本署名方式の性能を実験によって測定し，サーバ台数に比例して TPS を向上できること，NFT よりも消費電力量が少なく，スケールダウンした構成変更も可能であること，署名人数にかかわらず集約署名の検証に掛かる時間が概ね一定であることを示した．

デジタル署名技術によって NFT と同等以上の権利管理機能を実現したことで，より多くの企業・サービスがデジタルデータの権利管理ビジネスを開始できる可能性が高まる．今後の本格的な Web3 時代に向け，デジタルデータを安全・安心に活用できるよう研究を続けていきたい．

参考文献

[1] 内閣官房デジタル市場競争会議,”デジタル市場競争に係る中期展望レポート”, (2020).
[2] 自民党 デジタル社会推進本部, “NFT ホワイトペーパー(案) Web3.0 時代を見据えたわが国の NFT 戦略”, (2022).
[3] 矢羽野晶子,岡徳之, “たった 1 年で 2 万倍成長、NFT 市場の最新動向。急拡大の一方、電撃的買収や犯罪の増加も”, https://ampmedia.jp/2022/05/05/nft/, (2022).
[4] "EIP-721: Non-Fungible Token Standard". Ethereum Improvement Proposals, (2018).
[5] BASILE, Davide, et al. Enhancing blockchain-based processes with decentralized oracles. In: *International Conference on Business Process Management*. Springer, Cham, (2021).
[6] CALDARELLI, Giulio; ROSSIGNOLI, Cecilia; ZARDINI, Alessandro. Overcoming the blockchain oracle problem in the traceability of non-fungible products. *Sustainability*, (2020).
[7] KARAME, Ghassan O.; ANDROULAKI, Elli; CAPKUN, Srdjan. Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin. *Cryptology EPrint Archive*, (2012).
[8] WANG, Qin, et al. Non-fungible token (NFT): Overview, evaluation, opportunities and challenges. *arXiv preprint arXiv*:2105.07447, (2021).
[9] FENG, Siqi, et al. StateSnap: A State-Aware P2P Storage Network for Blockchain NFT Content Data. In: *International Conference on Algorithms and Architectures for Parallel Processing*. Springer, Cham, (2021).
[10] CAO, Bin, et al. Performance analysis and comparison of PoW, PoS and DAG based blockchains. *Digital Communications and Networks*, (2020).
[11] TRUBY, Jon, et al. Blockchain, climate damage, and death: Policy interventions to reduce the carbon emissions, mortality, and net-zero implications of non-fungible tokens and Bitcoin. *Energy Research & Social Science*, (2022).
[12] RISTENPART, Thomas; YILEK, Scott. The power of proofs-of-possession: Securing multiparty signatures against rogue-key attacks. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, Berlin, Heidelberg, (2007).
[13] MAXWELL, Gregory, et al. Simple schnorr multi-signatures with applications to bitcoin. *Designs, Codes and Cryptography*,(2019).
[14] BONEH, Dan; DRIJVERS, Manu; NEVEN, Gregory. Compact multi-signatures for smaller blockchains. In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, Cham, (2018).
[15] BELLARE, Mihir; NEVEN, Gregory. Multi-signatures in the plain public-key model and a general forking lemma. In: *Proceedings of the 13th ACM conference on Computer and communications security*. (2006).
[16] AMBROSIN, Moreno, et al. SANA: Secure and scalable aggregate network attestation. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*.(2016).
[17] LYSYANSKAYA, Anna, et al. Sequential aggregate signatures from trapdoor permutations. In: *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, Berlin, Heidelberg, (2004).
[18] KOJIMA, Rikuhiro, et al. A New Schnorr Multi-Signatures to Support Both Multiple Messages Signing and Key Aggregation. *Journal of Information Processing*, (2021).
[19] 野村総合研究所, "令和３年度エネルギー需給構造高度化対策に関する調査等事業（業務部門における更なる省エネの促進に向けた省エネ法関連制度に関する調査）報告書", (2022).