Adaptor SignatureとTrapdoor Commitmentを用いたVerifiable Credentialによる条件達成証明方式

張一凡[1,2,a)]　松浦 幹太[2]

概要：近年，自己主権型のユーザ属性証明技術として Verifiable Credential（VC）が注目されている．VC は発行者が対象者に対して静的な属性や状態を記載・証明する仕組みであり，既存方式では条件達成や達成時の追加情報といった動的な事実の証明は困難であった．たとえば，商品の保証期間を VC として発行することは可能だが，実際の使用開始日など行動に基づく情報を後から安全に組み込むことはできず，VC の再発行が必要となる．本研究では，動的な条件達成を安全に証明できる新たな VC 発行・検証手法を提案する．提案手法は，条件成立時に未完成署名を正式署名へ変換可能とする Adaptor Signature をベースに Trapdoor Commitment および Verifiable Data Registry（VDR）を統合的に利用する．これにより，商品の到着日から効力を持つ保証書など，ユーザ行動に応じた動的な VC を安全に発行できる．本手法により，VC の証明対象は記載内容だけではなく，VC 発行後の条件達成やその達成内容へと拡張できる．

キーワード：Verifiable Credential, 条件達成証明, アダプター署名，トラップドアコミットメント，ブロックチェーン

Condition-Based Verifiable Credential Issuance and Proofs Using Adaptor Signatures and Trapdoor Commitments

Iifan Tyou[1,2,a)]　Kanta Matsuura[2]

Abstract: Verifiable Credential (VC) have emerged as a self-sovereign framework for proving user attributes. While effective for static claims, existing schemes cannot natively prove dynamic facts such as condition fulfillment or state changes. For example, a VC can represent a warranty period but cannot securely incorporate the actual activation date. We propose a VC issuance and verification method that proves dynamic condition fulfillment by combining adaptor signatures, trapdoor commitments, and a blockchain-based Verifiable Data Registry (VDR). The design supports use cases such as warranties activated upon product delivery or conference passes validated upon attendance. This approach extends VC beyond static attributes to securely attest both the occurrence of a condition and related contextual details.

Keywords: Verifiable Credential, Proof of Condition Fulfillment, Adaptor Signature, Trapdoor Commitment, Blockchain

1. はじめに

近年，Verifiable Credential（VC）技術 [14] は，暗号的に保証されたデジタル証明書を用いて発行者 (Issuer)・所有者（Holder）・検証者 (Verifier) の三者間で安全かつプライバシー保護された属性証明を実現する手法として急速に普及している．W3C によってその標準仕様が整備され，政府発行 ID，学位，免許証など多様なユースケースでの導入が本格化している [12].

VC の典型的な利用フローでは，Issuer は Holder の属性情報とその情報へのデジタル署名を VC に記載することで保証し Holder に提供する．Holder は証明したい Verifier に

[1] NTT 株式会社 社会情報研究所
Social Informatics Laboratories, NTT
[2] 東京大学生産技術研究所
Institute of Industrial Science, The University of Tokyo
[a)] iifan.tyou@ntt.com

対して VC に署名を付けて提示し，Verifier は Holder の正しさを Holder のデジタル署名から，VC の正しさを Issuer の信頼性とそのデジタル署名に基づいて検証する．

このような VC 発行・利用では，静的に確定した属性情報の保証が一般的であり，動的な状態変化や実績に応じた VC の発行の検討が十分になされていなかった．具体的には，既存の VC 利用で Issuer は確認できた内容に対して VC を発行できる一方で，Issuer の状態確認を求めない**条件達成時の VC 発行**や**条件達成時の追加情報の保証**などへの拡張はほとんど考慮されていない．

本研究ではこれを打開するため，プレ VC を発行する Issuer と条件達成時の追加保証主体 Notary を分離し，一つの検証可能な VC を完成させることで，動的な条件達成証明とを目的とする．この目的で実現するユースケースの一例を以下に記す．

ユースケース例: 商品の保証 VC

家電などの製品保証として VC を払い出すことを考える．製造元（Issuer）は，売買契約成立時に製品シリアル番号や保証条件を含む仮の保証書（プレ VC）を発行し，顧客（Holder）に送付する．このプレ VC を正規の VC として完成させるためには使用開始年月日が必要であり，その開始日から保証期間が始まる．プレ VC を受け取った Holder は製品を受け取る際に郵送業者や小売店（Notary）から配送日時を記載してもらうことで，プレ VC を完成させて保証書 VC を得る．故障した際の修理業者（Verifier）は Issuer 指定の保証期間と Notary が示す使用開始日の双方を参照して，VC 検証を行い保証期間かを判断する．このユースケースでは Notary が配達日時を VC に追加することが VC 完成の条件となっており，その追加情報によって Issuer が発行する保証条件 Issuer が発行した保証条件に加えて，Notary による使用開始時期情報を含むことで，保証期間の正確な算定や第三者による検証が可能となる．

本研究が満たすべき要件は以下の通りである：

- **条件達成時の証明完成**：Issuer による再署名を行わずに，Notary による条件達成証明によって VC を完成できること．
- **メッセージ追加可能性**：署名を完成させるタイミングで情報を追記できること．
- **偽造不可能性**：攻撃者が Issuer の秘密鍵を持たない場合に未署名メッセージに対して有効な署名を作り出せず，かつ，Notary の秘密情報がない状態で条件達成時のメッセージ追加ができないこと．
- **既存インフラ互換性**：標準的な署名検証アルゴリズム ECDSA などとの互換性を保つこと．
- **性能**：モバイル端末や組み込み機器でも実行可能な計算コストであること．

1.1 既存手法の限界

提案するような証明者（Issuer），追加保証者（Notary）が分かれるような証明は，上記のユースケース以外にも契約での相互押印や参加証明など現実でも多く利用されている．Notary を追加することで証明を完成させるシンプルな実現方法として，Notary が新たな VC を発行する方式が考えられる．しかし，2 つの VC を示す場合 2 つの VC 間に暗号的な関連性をもたせづらく，Verifier に 2 つの発行者への信頼を求めてしまう．同時に VC 検証が 2 回必要になり検証コストが増加し，Issuer，Notary の署名方式に合わせた検証も必要となる．また，本来条件未達成の状態でも署名が検証可能なため，条件付き発行の制約が形骸化することで制約をかけた VC 発行ができなくなってしまう．

Issuer, Notary の二者による署名以外に高機能デジタル署名方式として近年注目されるアダプター署名（Adaptor Signature[3]）やオンライン・オフライン署名（Online/Offline Signatures[4], [13]）なども方式の実現には有効となる．詳細は 2 節で述べるが，いずれも想定するメッセージの追加が困難である．

1.2 提案方式と貢献

上記の課題を解決するため，本論文では，**Adaptor Signature** と **Trapdoor Commitment** および VC 検証で利用される記録台帳である Verifiable Data Registry（VDR）を用いて，条件達成時にのみ成立する VC 署名スキームを提案する．

本方式では，署名発行を行う Issuer が発行対象メッセージ m_I に対してプレ署名 (pre-signature) を生成する．プレ署名を受け取った Holder は，正式署名に変換するために条件を達成させ，その保証人である Notary の協力を得て達成時メッセージ m_N を追加したうえでプレ署名を正式署名に変換する．その際に，Trapdoor Commitment 手法を用いることで，署名メッセージ m_I とは別のメッセージ m_N への開示情報 open_N を生成し，保証書 VC に対する使用開始時刻を追加する．Trapdoor Commitment の特性から，Holder は open_N から秘密情報（Trapdoor td）を抽出できる．第三のメッセージに対する新たなコミットメント開示を防止し，実績証明を VDR に対して行う．この処理では Issuer はプレ署名発行後の処理には一切関係せず，Notary と Holder のみで署名を完成できる．

本論文の主な貢献は以下のとおりである．

(1) **メッセージを追加可能にした VC 発行プロトコル設計**　Adaptor Signature と Trapdoor Commitment を変形し，VDR と接続することで柔軟な VC 発行方式を提案

(2) **理論的安全性保証**　提案するオフライン VC 発行方式での Unforgeability，ユーザ情報の隠蔽性，情報追加の一意性に関する安全性を証明．

2. 関連研究

条件達成時にデジタル署名を自動的に完成させる方式として，主に Online ／ Offline 署名，マルチ署名／閾値署名，(Partially) Blind 署名などが提案されている．これらの方式はいずれも，事前生成された情報（プレ署名，タイムロック，コミットメントなど）と，後続の条件達成証拠を組み合わせて最終署名を生成する事ができる．

2.1 Online ／ Offline 署名

Even らが提案した Online ／ Offline 署名方式では，オフライン段階で大量のプレ署名を生成し，オンライン段階で高速な最終署名生成を可能とする [4]．Shamir らは Chameleon Hash（Trapdoor Hash）を利用した “hash-sign-switch” パラダイムを提案し，既存署名方式を効率的に Online ／ Offline に変換可能とした [13]．しかしこの方式では，事前署名時にメッセージを組み込むことが原理的に困難であり，条件達成に応じた追加情報（例：使用開始日や配送日時など）の埋め込みには不向きである．

2.2 マルチ署名／閾値署名および Blind 署名

マルチ署名および閾値署名では，複数の署名者が部分署名を生成し，条件達成時にオフラインで集約することで最終署名を得る方式であり，検証負荷の低減などの利点を有する [6], [8], [10]．しかし，プレ署名生成後の動的な情報追加には対応していない．Partially Blind Signature や Blind Adaptor Signature は，発行時にブラインド化された署名を行い，後でアンブラインドすることで最終署名を得る方式であるが，生成される内容は発行時に固定されており，動的追加には対応困難である [1]．

2.3 Adaptor Signature の最近の進展

Adaptor Signature (AS) は，条件付き署名完成の枠組みとして注目を集めている．Dai らは NP 関係に対して任意の NP 関係に対応するアダプタ署名の汎用構成を示したものの，適応後署名から Witness が漏洩する問題があった [2]．これに対し Liu らは，Witness を秘匿するセキュリティ定義を導入し，Trapdoor Commitment を組み込んだ Witness-Hiding Adaptor Signature を提案し，任意の NP 関係への適用を可能にした [9]．しかし，Adaptor Signature 単体でも情報の追加対応についての検討がなされていなかった．

2.4 本研究との違い

これらの方式はいずれも条件達成時に署名を自動完成させることは可能であるものの，本研究が要求する「**メッセージ追加可能性**」を満たしておらず，動的な条件達成情報の安全な組み込みには対応していない．本研究では，Adaptor Signature と Trapdoor Commitment を組み合わせ，その上で Verifiable Data Registry (VDR) を統合することで，条件達成時の情報追加と整合性保証の両立を実現する新しい VC 発行・検証方式を提案する．

3. 前提知識

本章では，本論文の提案方式を理解し，安全性証明を行うために必要な暗号プリミティブと記法を定義する．これら構成要素は提案プロトコルの設計と解析の基盤となる．

3.1 記法

- λ：セキュリティパラメータ．
- sk, vk：署名鍵および検証鍵．Issuer の鍵を $(\mathsf{sk}_I, \mathsf{vk}_I)$ と表す．
- M：署名対象メッセージ空間．m: 対象メッセージ
- $\tilde{\sigma}$：Adaptor Signature のプレ署名 (pre-signature)．
- σ：検証可能な通常デジタル署名．
- PPTA：確率的多項式時間アルゴリズム（Probabilistic Polynomial-Time Algorithm）．
- $\mathsf{negl}(\lambda)$：無視可能函数 $\mathsf{negl}(\lambda) < 1/\lambda^c$

3.2 デジタル署名スキーム

定義 1. 標準的なデジタル署名スキーム SIG は**署名者が署名鍵を元に署名行為を実施したことを証明**する暗号的手法である．

- $\mathsf{SIG.KG}(1^\lambda) \to (\mathsf{sk}, \mathsf{vk})$：安全性パラメータ λ を入力し，署名鍵（署名鍵）sk，検証鍵（公開鍵）vk のペアを生成．
- $\mathsf{SIG.Sign}(\mathsf{sk}, m) \to \sigma$：メッセージ m と署名鍵 sk を入力し，署名 σ を生成．
- $\mathsf{SIG.Ver}(\mathsf{vk}, m, \sigma) \to \{0,1\}$：検証鍵 vk，署名対象メッセージ m，署名 σ を入力し，有効な署名 1 か否 0 かを返す．

デジタル署名の要件として正当性（**Correctness**）では検証鍵 vk に対応した正しいメッセージ m と署名 σ の組み合わせに対して SIG.Ver は必ず 1 を返すことが求められる．また，偽造不可能性（**Unforgeability**）としては **EUF-CMA**(Unforgeability under Chosen-Message Attack) などの安全性定義に沿った偽造の不可能性への保証が求められる．その実装として，シュノア署名や ECDSA 署名，BBS 署名など多様なプロトコル・実装が存在する．

3.3 NP 言語とハードリレーション

多項式時間で検証可能なバイナリ関係 $R(W, w) \subseteq \{0,1\}^* \times \{0,1\}^*$ を用いて, NP 言語 L は $L = \{W \in \{0,1\}^* \mid \exists w \in \{0,1\}^* \text{ s.t. } (W, w) \in R\}$ により定義される．ここで W を ステートメント (statement) と呼び, w を $(W, w) \in R$ の証拠（witness）と呼ぶ．

関係 R のうち，statement から witness を多項式時間では見つけにくい一方で，与えられた witness が statement に対して R に含まれるかを多項式時間で検証可能なものをハードリレーション（Hard Relation）と呼ぶ．厳密には関係 $R(W,w)$ がハードリレーションであるとは，任意の PPTA の攻撃者 $\mathcal{A}$ が関係 R を用いて，秘密情報として成立する w' を抽出できる確率，つまり $\mathsf{Succ}_{R,\mathcal{A}}(\lambda) := Pr[(W,w) \in R; w' \leftarrow \mathcal{A}(R,W) : (W,w') \in R]$ が無視できることを意味する．

NP 言語に対応する「決定問題」をまとめて NP 問題と呼び，NP 完全問題は NP 問題の中でも最も難しい部類にある．暗号でよく使われる，離散対数問題（Discrete Logarithm Problem： DLP）や楕円曲線上の離散対数問題（Elliptic Curve DLP : ECDLP）は有限体や楕円曲線の代数的構造に依存した NP 問題であるが，NP 完全ではない．一方，NP 完全問題であるハミルトン閉路問題（Hamiltonian Cycle Problem：HCP）は任意の NP 関係をカバーできるため，より一般的な応用が可能となる．

3.4 コミットメントスキーム

定義 2. コミットメント（Commitment） スキームは，あらかじめ**情報を隠しつつ後で確定的に開示できる**暗号的手法であり，以下のアルゴリズムを持つ：

- $\mathsf{CS.Gen}(1^\lambda) \to \mathsf{ck}$：セキュリティパラメータ λ を入力し，コミットメント鍵 ck を生成.
- $\mathsf{CS.Com}(\mathsf{ck}, m) \to (\mathsf{com}, \mathsf{open})$：コミットメント鍵 ck とメッセージ m を入力し，コミットメント com と開示情報である open を生成.
- $\mathsf{CS.Ver}(\mathsf{ck}, \mathsf{com}, m, \mathsf{open}) \to \{0,1\}$：コミットメント鍵 ck とコミット対象メッセージ m，コミットメント com と開示情報 open を入力し，com, open が m と整合する 1 か否 0 かを返す.

Commitment スキームの要件として正当性（**Correctness**）では正しい $\mathsf{ck}, \mathsf{com}, m, \mathsf{open}$ の組に対して CS.Verify は必ず 1 を返すことが求められる．また，束縛性（**Binding**）では一度コミットメント com を作成すると，異なるメッセージ $m' \neq m$ および $\mathsf{open}' \neq \mathsf{open}$ を対応づけることは計算量的にできないこと，および隠蔽性（**Hiding**）としてコミットメント com 単体からはコミットしたメッセージ m の情報が計算量的に漏洩しないことを満たす必要がある．

3.5 トラップドア付きコミットメントスキーム

定義 3. トラップドア付きコミットメントスキーム（Trapdoor Commitment）は，Commitment スキームでのコミットメント com を Trapdoor によって 生成時とは別のメッセージ m_T，開示情報 open_T で検証可能にしたコミットメント方式であり，以下のアルゴリズムを持つ [5]：

- $\mathsf{TC.Gen}(1^\lambda) \to (\mathsf{ck}, \mathsf{td})$：セキュリティパラメータ λ を入力し，コミットメント鍵 ck と Trapdoor td を生成.
- $\mathsf{TC.Com}(\mathsf{ck}, m_0) \to (\mathsf{com}, \mathsf{open}_0)$：コミットメント鍵 ck とコミットする情報 m_0 を入力し，コミットメント com と Trapdoor を用いない初期開示情報 open_0 を生成.
- $\mathsf{TC.TdOpen}(\mathsf{td}, \mathsf{com}, m_T, m_0, \mathsf{open}_0) \to \mathsf{open}_T$：Trapdoor td，コミットメント com，新たに開示情報を生成したいメッセージ m_T と過去のメッセージ・開示情報セット m_0, open_0 を入力し，生成済みのコミットメント com に対応した新たな開示情報 open_T を生成.
- $\mathsf{TC.Ver}(\mathsf{ck}, \mathsf{com}, m_i, \mathsf{open}_i) \to \{0,1\}$：コミットメント鍵 ck とコミットメント com，コミットする情報 m_i，開示情報 open_i を入力し，コミットメント com が開示された情報 m_i と整合するかを返す.
- $\mathsf{TC.Ext}(\mathsf{ck}, \mathsf{com}, m_i, \mathsf{open}_i, m, \mathsf{open}) \to \mathsf{td}$：コミットメント鍵 ck とコミットメント com，コミットした 2 つのメッセージ m_i, m とそれぞれの開示情報 $\mathsf{open}_i, \mathsf{open}$ を入力し，ck と整合する Trapdoor td を返す.

Trapdoor Commitmen の要件としては Commitment スキームと同様に **Correctness** と td を知らない攻撃者に対して **Binding** 性と **Hiding** 性を保持しつつ，Trapdoor によって任意のメッセージ m_i, m_j に対して生成される $(\mathsf{open}_i, \mathsf{com}_i), (\mathsf{open}_j, \mathsf{com}_j)$ が識別不可である **Equivocability** と，TC.Ext で定義される com に対する衝突したメッセージ m_i, m_0 と $\mathsf{open}_i, \mathsf{open}_0$ から Trapdoor td を取り出す抽出可能性（**Extractability**）を満たす必要がある．

3.6 アダプター署名スキーム

定義 4. アダプター署名（Adaptor Signature）はハードリレーション $(W,w) \in R$ に関連付けられた署名方式である [3]．プレ署名と呼ばれる未完成署名 $\tilde{\sigma}$ を発行し，その後所定の秘密（witness w）を用いて本来の署名を完成できる機構を有する．

(1) $\mathsf{AS.Gen}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$：安全性パラメータ λ を入力し，署名鍵 sk，検証鍵 vk のペアを生成する.

(2) $\mathsf{AS.pSign}(\mathsf{sk}, m, W) \to \tilde{\sigma}$：メッセージ m と署名鍵 sk，$(W,w) \in R$ である statement W を入力し，プレ署名 $\tilde{\sigma}$ の生成.

(3) $\mathsf{AS.pVer}(\mathsf{vk}, m, W, \tilde{\sigma}) \to \{0,1\}$：署名者の検証鍵 vk，メッセージ m，$(W,w) \in R$ の statement である W，プレ署名 $\tilde{\sigma}$ を入力し，プレ署名検証として有効なプレ署名 1 か否 0 かを返す.

(4) $\mathsf{AS.Adapt}(\mathsf{vk}, m, \tilde{\sigma}, w) \to \sigma$：プレ署名者の検証鍵 vk，メッセージ m，$(W,w) \in R$ の witness w，プレ署名 $\tilde{\sigma}$ を入力し，SIG.Ver により検証できる完全なデジタル署名 σ を生成する.

(5) $\mathsf{AS.Ext}(\mathsf{vk}, m, W, \tilde{\sigma}, \sigma) \to \perp$ or w：プレ署名者の検証鍵 vk，メッセージ m，statement W，プレ署名 $\tilde{\sigma}$ と署名 σ を入力し，正しいプレ署名・署名ペアであった場合は witness w を出力し，そうでない場合は $\perp$ を出力．

Adaptor Signature もデジタル署名で求められる性質を満たすことが求められ，追加として AS.Adapt，AS.Ext の実現可能性が求められる．Liu らが提案する Trapdoor Commitment を用いた Adaptor Signature 構成 [9] では Trapdoor Commitment の各機能と既存の任意デジタル署名 SIG を利用する．その効果として，「事前署名と署名のどちらか一方だけでは witness を導出できない witness-hiding」を確保し，「任意署名方式に対応できる汎用方式」かつ「任意の NP 言語を利用できるように HCP に基づく方式」を実現できる．

3.7 VDR・ブロックチェーン

VC で参照される Verifiable Data Registry（VDR）では一般的に DID と公開鍵の対応情報や失効情報が記録されるデータレジストリを指す．一般的に VDR は正しく運用されており，改ざんされないものと想定する．集権的な主体によって VDR が運用される場合，運用主体と Issuer あるいは Holder の関係から記録データを偽装する可能性が生じる．群衆監視により，この様な偽装の懸念を払拭するために，ブロックチェーンを用いる VDR[7] も広く利用されており，この様な実装では VDR 記録に対する順序に対してブロックチェーンの性質が付加される．

ブロックチェーンは複数の運用主体がコンセンサスアルゴリズムを用いて同一の台帳を維持するシステムである．正しく運用されているブロックチェーンとして，Pass らの研究 [11] を参照して，Consistency と Liveness が満たされることを想定する．この 2 つの性質を満たすブロックチェーン型の VDR では，概念的には「一定期間前のトランザクションは必ずすべての正直な台帳ノードに一致して記録されること」と「改ざんやノード間の順番の差が無いこと」が保証されるものと考える．

本稿で VDR として説明するときは，任意の検証者が信頼できる運用主体による運用か，上記のブロックチェーンの性質を満たす運用がなされている，改ざんや順序の不一致が発生しないものを前提とする．

4. 提案方式

本章では，Trapdoor Commitment を組み合わせた提案方式を説明する前に，前提として Adaptor Signature 単体をそのまま利用した条件付き VC の発行を解説する．その後，この手法に Trapdoor Commitment と VDR 記録を導入した提案方式を紹介する．VC 発行の参加者としては一般的な**発行者:Issuer**，**保有者:Holder**，**検証者:Verifier** と VDR に加えて，新たに**補助者:Notary** を追加する．

4.1 Adaptor Signature を用いた VC 発行

Issuer が事前に生成したプレ署名 $\tilde{\sigma}$ を Holder に渡し，条件達成時に Notary が witness w を Holder に提供することで，Holder は AS.Adapt により最終署名を得るまでの一連の処理を示す．

- **事前準備** Issuer は $\mathsf{ASDVC.Setup}_{Issuer}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$ として Adaptor Signature の AS.Gen によって生成される鍵を持つ．
  Notary は $\mathsf{ASDVC.Setup}_{Notary}(R) \to (W, w)$ でハードリレーション R から Witness w とそれに対応する Statement W を生成し，Statement W を Issuer に共有する．
- **条件未達成署名生成** $\mathsf{ASDVC.pSign}(\mathsf{sk}, m, W) \to \tilde{\sigma}$：Issuer は保有する秘密鍵 sk と受領した Statement W を用いて既存のアダプター署名方式を用いたプレ署名 $\tilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{sk}, m, W)$ を生成して Holder に配布する．
- **条件未達成署名検証** $\mathsf{ASDVC.pVer}(\mathsf{vk}, m, W, \tilde{\sigma}) \to \{0, 1\}$：Holder は受領した $\tilde{\sigma}$ が正しく生成されたことを，AS.pVer 検証を用いて Issuer の公開鍵 vk と Statement W を使って検証する．そのため，$\mathsf{ASDVC.pVer} = \mathsf{AS.pVer}$ をそのまま出力する．
- **条件達成時の署名完成** $\mathsf{ASDVC.Adapt}(\mathsf{vk}, m, \tilde{\sigma}, w) \to \sigma$：Holder が条件達成時に Notary が保有する witness w を用いて，Holder の提示するプレ署名に対して AS.Adapt を実施することで，プレ署名 $\tilde{\sigma}$ を検証可能な通常署名 σ に変換（Adapt）する．その際に VDR に条件達成時の追加情報を保管し Verifier から参照可能にすることで，追加情報の検証可能性も確保できる．処理としては，$\mathsf{ASDVC.Adapt} = \mathsf{AS.Adapt}$ によって実現されるが，後述する Trapdoor Commitment を用いる方式と違い，AS.Adapt では VC のメッセージ m を入力する必要があるため，**Notary に対する VC 記述の秘匿性を確保できない**．
- **最終署名検証** $\mathsf{ASDVC.Ver}(\mathsf{vk}, m, \sigma) \to \{0, 1\}$：Verifier は署名 σ が Issuer の検証鍵 vk とメッセージ m に対して成立するかを検証する．ここはデジタル署名の検証に従い，$\mathsf{ASDVC.Ver} = \mathsf{SIG.Ver}$ で実現される．これは後述する方式と違い，追加の Trapdoor Commitment の実装が必要ない点で**汎用性が高い**．

上記の処理により条件達成時にその保証を行う Notary が Witness w を用いて VC の完成が可能になる．この手法は任意の Adaptor Signature に対して有効であり，ECDSA ベースの Adaptor Signature に適用することで最終的には ECDSA で検証可能な署名を持つ VC が生成できる．この方式はシンプルに要件を実現できている一方で，Notary

に対して VC のメッセージを開示する必要がありプライバシー課題が生じる．また，AS.Ext など Adaptor Signature の機能を十分に活かせていない．

4.2 Trapdoor Commitment と VDR を追加した VC 発行

事前に Issuer が生成されたプレ署名に対して，Notary が条件達成時にメッセージ m_N を追加して適応（Adapt）し，Holder が最終署名を得て，Verifier が検証できるまでの一連の処理を示す．メッセージの追加では Trapdoor Commitment を用いて，事前のコミットメント com に対応した開示情報 open を生成する．

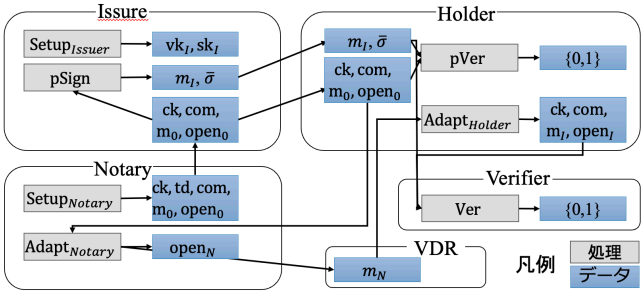

図 1 Trapdoor Commitment を用いた提案スキーム

- **事前準備** Issuer は $\mathsf{Setup}_{Issuer}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$ で検証鍵，署名鍵を生成する．

$$(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{SIG.KG}(1^\lambda)$$

Notary は $\mathsf{Setup}_{Notary}(1^\lambda, R) \to (\mathsf{ck}, \mathsf{td}, \mathsf{com}, m_0, \mathsf{open}_0)$ でハードリレーション R から Witness w に相当する Trapdoor td とそれに対応する Statement W に相当する Commitment 鍵 ck および TC.Ver に利用できるランダムのメッセージ m_0 とそれに対するコミットメント，開示情報である $\mathsf{com}, \mathsf{open}_0$ を生成する．そこから $w, \mathsf{com}, m_0, \mathsf{open}_0$ を Issuer へ要求に応じて送付する．ここで Liu らの構成 [9] に合わせ，$(W, w) = (\mathsf{ck}, \mathsf{td})$ として利用する．

$$(W, w) = (\mathsf{ck}, \mathsf{td}) \leftarrow Sample(R)$$
$$m_0 \xleftarrow{?} M$$
$$\mathsf{com}, \mathsf{open}_0 \leftarrow \mathsf{TC.Com}(\mathsf{ck}, m_0).$$

- **条件未達成署名生成** Issuer は Holder からのリクエストにあわせて $\mathsf{pSign}(\mathsf{sk}, m_I, \mathsf{ck}) \to \tilde{\sigma}$ で Holder へメッセージ m_I と，m_I 対応した条件未達成のプレ署名 $\tilde{\sigma}$ を発行し，Holder へ送付する．ここで重要なことは $\tilde{\sigma}$ に含まれる開示情報 open_0 は署名メッセージ m_I ではなく，事前の乱数 m_0 に対応していることである．

$$\bar{\sigma} \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}, (m_I|\mathsf{ck}|\mathsf{com})).$$
$$\tilde{\sigma} \leftarrow (\bar{\sigma}, \mathsf{ck}, \mathsf{com}, \mathsf{open}_0, m_0)$$

- **条件未達成署名検証** Holder は $\mathsf{pVer}(\mathsf{vk}, m_I, \tilde{\sigma}) \to \{0,1\}$ で Issuer から受領した $m_I, \tilde{\sigma}$ が正しく生成され，条件達成時に Adapt により本署名に変換できることを下記より検証する．ここで重要なことは TC.Ver で検証できるということは，ck, com に対して新たなメッセージと開示情報が得られたとき，Trapdoor Commitment の抽出可能性によって Trapdoor td を得られると確認できることである．

$$\mathsf{SIG.Ver}(\mathsf{vk}_I, (m_I|\mathsf{ck}|\mathsf{com}), \bar{\sigma}) \stackrel{?}{=} 1$$
$$\mathsf{TC.Ver}(\mathsf{ck}, \mathsf{com}, m_0, \mathsf{open}_0) \stackrel{?}{=} 1.$$

- **条件達成時の署名完成** Notary は $\mathsf{pAdapt}_{Notary}(\mathsf{com}, \mathsf{td}) \to \sigma$ で Holder が条件達成した際に，Holder が提示した com に対してその事実を確認して追加情報 m_N を生成し，保有する Trapdoor td を用いて m_N に対するオープン open_N を生成し，VDR に $(\mathsf{com}, m_N, \mathsf{open}_N)$ を記録する．VDR の記録完了後にそのリンクを Holder に返送する．Notary は VDR に $(\mathsf{com}, m_N, \mathsf{open}_N)$ を記録することで Holder が td 抽出後に新たな m'_N を作成しても，VDR 上の**最初の** com に対するメッセージにできないようにしている．同時に Notary は m_I を受け取らない事によって Holder のプライバシーを確保でき，m_I に対する open_I も Notary では作成しない，

$$\mathsf{open}_N \leftarrow \mathsf{TC.TdOpen}(\mathsf{td}, \mathsf{com}, m_N, m_0, \mathsf{open}_0)$$

Holder は $\mathsf{pAdapt}_{Holder}(\mathsf{open}_N, m_N) \to \mathsf{open}_I$ で VDR に開示された追加情報 m_N と open_N より TC.Ext によって Trapdoor td を抽出する．その後 td を用いて Issuer が署名したメッセージ m_I に対する open_I を生成し，検証可能な最終署名 σ を保管する．

$$\mathsf{td} \leftarrow \mathsf{TC.Ext}(\mathsf{ck}, \mathsf{com}, m_0, \mathsf{open}_0, m_N, \mathsf{open}_N)$$
$$\mathsf{open}_I \leftarrow \mathsf{TC.TdOpen}(\mathsf{td}, \mathsf{com}, m_I, m_0, \mathsf{open}_0)$$
$$\sigma \leftarrow (\bar{\sigma}, \mathsf{ck}, \mathsf{com}, m_I, \mathsf{open}_I).$$

- **最終署名検証** $\mathsf{Ver}(\mathsf{vk}, m_I, \sigma) \to \{0,1\}$：Verifier は VC の記録メッセージ m_I に対して署名 σ が Issuer の検証鍵 vk で検証可能かに加え，Trapdoor Commitment の正しさを検証するとで条件達成済みかを知ることができる．また，VDR 上でメッセージの**最初の** com, m_N セットを検証することで署名の正しさに加え，Holder が条件達成時の追加情報も確認できる．

$$\mathsf{SIG.Ver}(\mathsf{vk}, (m_I|\mathsf{ck}|\mathsf{com}), \bar{\sigma}) \stackrel{?}{=} 1$$
$$\mathsf{TC.Ver}(\mathsf{ck}, \mathsf{com}, m_I, \mathsf{open}_I) \stackrel{?}{=} 1.$$

上記の処理により条件達成検証を行う Notary から提供される条件達成情報 m_N とその開示情報 open_N を用いた VC の完成が可能になる．この手法は Liu らの手法をベースに Notary からの開示情報の展開フローを変更したプロトコルであり，SIG として任意の既存署名スキームに対して利用可能である．例として，ECDSA 署名を利用することで既存の VC 検証プロトコルに対して，VDR 検証と Commitment Scheme 検証を追加した形となり，既存プロトコルの変更も少なくおさえられる．

提案方式では先述した Adaptor Signature のみを用いた VC 発行と比べて Notary は open_N, m_N を生成するときに Holder の情報を必要としないため，Holder の Notary に対する m_I の秘匿性も確保できている．

提案手法のさらなる機密性向上も改変として考えられる．例えば，VDR に対する開示 m_N を $Hash(m_N)$ として，m_N を Holder に送り返すことで第三者に対して m_N を検証できなくすることもできる．また，com に対して Notary の署名をつけることなども考えられる．本稿ではシンプルな実用例として Trapdoor Commitment を用いることで VC を作成する一例を示したものである．

4.3 プロトコルが満たす要件

チケット発行と，条件達成時に証明書をオフライン発行するシステムとして，システムが満たす要件を整理する．

- **条件による証明完成**：Issuer が直接証明しなくても，Issuer が委託する Notary が pAdapt_{Notary} によって条件達成を証明することで，Verifier は Holder が条件を達成できた事実を検証できる．
- **メッセージ追加可能性**：Notary が条件達成とその達成情報を記録することで，Verifier は Holder が条件をどの程度達成できたのかなど追加の情報を Issuer の証明のもとで検証できる．
- **既存インフラ互換性**：既存の ECDSA などのデジタル署名と VDR による時系列的なデータ証明を利用し，条件の達成を検証可能になる．追加としてコミットメントスキームを導入することで，条件達成時情報の検証が可能になる．
- **性能**：既存の離散対数問題を NP 関係として利用する既存実装[*1] から処理時間がマイクロ秒オーダーと短く処理負荷も軽量となるであることが確認できた．

安全性について満たす性質については 5 にて述べる．

5. 安全性の証明

5.1 想定する脅威モデル

提案する VC 発行プロトコルのセキュリティ特性で考慮する各参加者の行動と脅威モデルを定義する．本研究では，攻撃者が事前にどの役割を侵害するかを決定する静的侵害モデルを想定する．この脅威モデルにおいて，**Issuer** は署名責任を負う立場であるため攻撃を考慮せずに忠実に処理を実行するものと考える．**Notary** はいわば事実の証明者の立場であり，プロトコル仕様に従って動作するが受信した情報から Holder に関わる情報を取得しようとするものと考える．**Holder** は自身に対する認証情報を偽造したり複数のアクセス権限を抽出しようとする目的で任意の行動を取る可能性があるものと考える．

5.2 各要件に対する安全性証明

安全性証明では詳細な攻撃定義については一部紙面の都合で割愛し，簡易的な記載に留める．

正当性（Correctness）

定理 1 (Correctness)**.** 正直な Issuer,Holder,Notary がプロトコルを正しく処理を実行すれば，正直な Verifier は常に最終署名 σ を受理できる．つまり，デジタル署名方式 SIG，Trapdoor Commitment TC が正当性を満たすならば，提案方式の最終署名 Σ は正当性を満たす．

証明. デジタル署名方式 SIG が正当性を満たすため，正直な仮定する Issuer から生成される署名 $\bar{\sigma}$ は常に，$\mathsf{SIG.Ver}(\mathsf{vk}_I, (m_I|\mathsf{ck}|\mathsf{com}), \mathsf{SIG.Sign}(\mathsf{sk}_I, (m_I|\mathsf{ck}|\mathsf{com}))) = 1$ となる．また，Trapdoor Commitment スキーム が抽出可能性と正当性を満たすため，正直な Notary から com に対する 2 つのメッセージ，開示情報 $(m_0, \mathsf{open}_0), (m_N, \mathsf{open}_N)$ を受け取る Holder は td を抽出できる．正しく抽出された td を元に m_I に対して生成された新しい開示情報 $open_I$ は Trapdoor Commitment スキームの正当性に従い，$\mathsf{TC.Ver}(\mathsf{ck}, \mathsf{com}, m_I, \mathsf{open}_I) = 1$ を満たす． □

安全性定義：偽造不可能性 (Unforgeability).

定義 5. 任意の PPTA 攻撃者 $\mathcal{A}$ に対し，攻撃成功確率を

$$\mathrm{Succ}^{uf}_{\mathcal{A}} := \Pr\left[\begin{array}{l}(\mathsf{ck}, \mathsf{com}, m_0, \mathsf{open}_0) \leftarrow \mathsf{Setup}_{Notary}(1^\lambda),\\ (\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Setup}_{Issuer}(1^\lambda),\\ (m, \sigma) \leftarrow \mathcal{A}^{\mathsf{pSign}(\cdot)}(\mathsf{vk}, \mathsf{ck}, \mathsf{com}, m_0, \mathsf{open}_0);\\ m^* \notin \mathcal{Q} \ \wedge \ \mathsf{Ver}(\mathsf{vk}_I, m, \sigma) = 1\end{array}\right]$$

と定義する．ここで，$\mathcal{A}$ は任意の m について $\mathsf{pSign}(m)$ を問い合わせ可能で，オラクルが応答した $\bar{\sigma}$ を得る．$\mathcal{Q}$ は $\mathcal{A}$ が pSign オラクルに問い合わせた全メッセージの集合である．この成功確率 $\mathrm{Succ}^{uf}_{\mathcal{A}} \leq \mathsf{negl}(\lambda)$ となる場合に偽造不可能性を満たすと定義する．

定理 2 (Unforgeability)**.** デジタル署名方式 SIG が EUF-CMA 安全で，Trapdoor Commitment TC が Binding 性を満たすならば，提案方式は偽造不可能性を満たす．

証明. $\mathcal{A}$ が任意のメッセージと偽造署名ペア $(m^*_I, \bar{\sigma}^*)$ を生成するために署名方式 SIG の EUF-CMA 安全性を破る必要がある．プレ署名 pSign で $\bar{\sigma}$ を取得する場合でも，署名方式

[*1] https://github.com/hl-tang/generic-adaptor-signature-from-trapdoor-commitment/

として SIG をそのまま用いているため，攻撃者は署名偽装に成功できるなら，ベースとする署名方式の EUF-CMA 安全性を破ることになり，その帰着により $\tilde{\sigma}$ 及び σ いずれの偽造の成功確率も Negligible となる．次に，プレ署名を受け取った場合に m_I を変えずにそのオープン $open_I$ を生成することを考える．ここで攻撃者は m_0 に対して生成されたコミット $com, open_0$ を元に（Notary の助けを得る前に）トラップドア td を知らずに $\mathsf{TC.Ver}(\mathsf{ck}, \mathsf{com}, m_i, \mathsf{open}_i) \to 0/1$ となる $open_i$ を生成する必要がある．トラップドア td がない場合の trapdoor commitment の Binding 性により，同じ com に対して 2 つの有効なオープンを生成できる確率は Negligible であるため，この様な攻撃者は $open_i$ の偽装ができない．結果として，EUF-CMA 証明の偽装成功確率 $\mathrm{Succ}_{\mathsf{SIG}}^{\mathrm{EUF-CMA}}$ と Trapdoor Commitment の Binding 性を破る確率 $\mathrm{Succ}_{\mathsf{TC}}^{\mathrm{Binding}}$ を用いて，

$$\mathrm{Succ}_{\mathcal{A}}^{uf} \leq \mathrm{Succ}_{\mathsf{SIG}}^{\mathrm{EUF-CMA}} + \mathrm{Succ}_{\mathsf{TC}}^{\mathrm{Binding}} \leq \mathsf{negl}$$

となり，偽造不可能性を確保できる． □

コミットメントの隠蔽性（Hiding）

定理 3 (Hiding)**.** Notary はコミット com から m_I を推測できない．

証明. 一般の Trapdoor Commitment は standard assumptions の下で，少なくとも計算上の隠蔽を満たすため，Adapt で Holder が Notary に対して com を送信するだけでは，Notary は自分が生成する m_N を知ることができるが，m_I にあるユーザ情報を特定できない． □

情報追加の一意性（One-Shot Usage）

定理 4 (One-Shot Usage)**.** Verifier が検証可能な条件達成時の追加情報 m_N は Notary が一度決めると攻撃者が追加・改変できない．

証明. Notary が $(\mathsf{com}, m_N, \mathsf{open}_N)$ を改変できず，時系列ソートされている台帳に記録してから，台帳を経由して攻撃者である Holder 情報を転送する．VDR が変更不可能で時系列ソートされている限り，正直な Verifier は検証ロジックに従い最初の (com, m_N) 組を参照するため，Holder が TC.Ext によってトラップドア td を入手して，新たな条件達成時の追加情報を $(\mathsf{com}, m_N")$ を VDR に公開しても Verifier からは参照されない． □

6. まとめ

本稿では，Issuer が事前に付与するメッセージだけではなく，条件達成とその時に動的に変動する追加情報をも検証可能にする VC 発行について提案を行った．提案手法を用いることで，事前にメッセージが確定した静的な VC 発行のみならず，Notary が提示する動的な条件達成を組み合わせた VC 発行と実績証明が可能になる．これによって，例として記載した保証書と利用開始のユースケースに加え．チケットと参加証明など事前設定した条件達成を伴う多くのユースケース VC 利用の幅を広げられると期待する．

今後の課題として，ウォレット環境での動作など実用に向けた検討が必要と考えられる．また，プロトコルの拡張としては，Trapdoor Commitment の実装と NP 言語のさらなる活用，ゼロ知識証明などによる VDR の必要性の回避などが期待される．また，耐量子計算機暗号での実現なども，今後の長期運用を想定した際に課題となるだろう．

謝辞．本研究の一部は，JST 戦略的創造研究推進事業（CREST）JPMJCR22M1 の支援を受けたものである．内容議論に参加された NTT 社会情報研究所のオクタビオさんはじめ関係者に感謝する．

参考文献

[1] Abe, M. and Okamoto, T.: Provably secure partially blind signatures, Annual international cryptology conference, Springer, pp. 271–286 (2000).
[2] Dai, W., Okamoto, T. and Yamamoto, G.: Stronger Security and Generic Constructions for Adaptor Signatures, INDOCRYPT 2022 (2022).
[3] Erwig, F., Müller, J. and Seiler, D.: Generic Adaptor Signatures from Identification Schemes, Proceedings of PKC 2021, Springer (2021).
[4] Even, S., Goldreich, O. and Micali, S.: On-line/off-line digital signatures, Conference on the Theory and Application of Cryptology, Springer, pp. 263–275 (1989).
[5] Fischlin, M.: Trapdoor commitment schemes and their applications, PhD Thesis, Frankfurt (Main), Univ., Diss., 2001 (2001).
[6] Gennaro, R. and Goldfeder, S.: Fast multiparty threshold ECDSA with fast trustless setup, Proceedings of the 2018 ACM SIGSAC conference on computer and communications security, pp. 1179–1194 (2018).
[7] Hyperledger-Foundation: Hyperledger Indy, `https://www.lfdecentralizedtrust.org/projects/hyperleger-indy`.
[8] Lindell, Y.: Fast secure two-party ECDSA signing, Journal of Cryptology, Vol. 34, No. 4, p. 44 (2021).
[9] Liu, X., Tzannetos, I. and Zikas, V.: Adaptor Signatures: New Security Definition and A Generic Construction for NP Relations, ASIACRYPT 2024 (2024).
[10] Nick, J., Ruffing, T. and Seurin, Y.: MuSig2: Simple two-round Schnorr multi-signatures, Annual international cryptology conference, Springer, pp. 189–221 (2021).
[11] Pass, R., Seeman, L. and Shelat, A.: Analysis of the Blockchain Protocol in Asynchronous Networks, Advances in Cryptology – EUROCRYPT 2017 (2017).
[12] Sedlmeir, J., Smethurst, R., Rieger, A. and Fridgen, G.: Digital identities and verifiable credentials, Business & Information Systems Engineering, Vol. 63, No. 5, pp. 603–613 (2021).
[13] Shamir, A. and Tauman, Y.: Improved online/offline signature schemes, Annual International Cryptology Conference, Springer, pp. 355–367 (2001).
[14] W3C: Verifiable Credentials Data Model v2.0, `https://www.w3.org/TR/vc-data-model-2.0/`.