

匿名性を持つ属性証明書発行シーケンスの形式検証

児保 亮樹^{1,a)} 渡部 翔^{1,b)} 米山 一樹^{1,c)} 石井 龍² 林田 淳一郎² 永井 達也² 野村 健太²
齋藤 恆和² 高田 雄太² 神蘭 雅紀²

概要：属性認証とは、ユーザの年齢や性別などの属性情報に基づきシステムのアクセス制御などを行う認証方式である。石井ら (ICSS 研究会 2025 年 3 月) は属性情報を発行機関に対し秘匿した状態で属性証明書を発行するためのブラインド署名・ハッシュ木を用いた証明書発行シーケンスを提案し、OpenSSL 3.4.0 による概念実証を行った。しかし、形式的な安全性評価は行われていない。本稿では、彼らの方式のうちブラインド署名を用いた証明書申請・発行シーケンスの安全性について、暗号プロトコルの自動検証ツール ProVerif により形式検証を行う。我々は属性管理機関が証明書発行機関以外から属性証明書を発行されないこと、証明書発行機関が属性管理機関以外に対して属性証明書を発行しないこと（なりすまし攻撃耐性）、および悪意のある証明書発行機関に対して属性情報を明らかにしないこと（秘匿性）を検証する。検証結果として、石井らの方式は上記の安全性要件をすべて満たすことを示す。

キーワード：フォーマルメソッド、属性認証、ブラインド署名

Formal Verification of Sequences of Attribute Certificate Issuance with Anonymity

YOSHIKI KOYASU^{1,a)} KAKERU WATANABE^{1,b)} KAZUKI YONEYAMA^{1,c)} RYU ISHII² JUNICHIRO HAYATA²
TATSUYA NAGAI² KENTA NOMURA² TSUNEKAZU SAITO² YUTA TAKATA² MASAKI KAMIZONO²

Abstract: Attribute authentication is a method that controls system access based on user attribute information such as age and gender. Ishii et al. (ICSS research meeting, March 2025) proposed a certificate issuance sequence using blind signatures and hash trees to issue attribute certificates while keeping the attribute information private from the issuer, and they conducted a proof of concept using OpenSSL 3.4.0. However, no formal security analysis has been shown. In this paper, we formally verify the security of their proposed certificate application and issuance sequence, which uses blind signatures, using the automated cryptographic protocol verification tool ProVerif. We verify authenticity such that the attribute manager is not issued an attribute certificate from anyone other than the certificate issuing authority, and the certificate issuer does not issue an attribute certificate to anyone other than the attribute manager, and secrecy such that the attribute information is not disclosed to the malicious certificate issuer. As a result of the verification, we show that Ishii et al.'s scheme satisfies all of the above security requirements.

Keywords: formal methods, attribute authentication, blind signature

1. はじめに

1.1 背景

属性認証は、ユーザの年齢や資格、身体情報といった属性情報に基づいて、サービスの提供可否やシステムのアクセス制御を行う認証方式である。近年、属性情報の利活用に

¹ 茨城大学

Ibaraki University

² デロイト トーマツ サイバー合同会社

Deloitte Tohmatsu Cyber LLC

a) 22t4034a@vc.ibaraki.ac.jp

b) 24nm771g@vc.ibaraki.ac.jp

c) kazuki.yoneyama.sec@vc.ibaraki.ac.jp

ついて社会的な需要が高まっている。国内での取り組みでは、デジタル庁が国家資格等に係る各種申請手続のオンライン化や資格情報の連携等のデジタル化を進めている [1]。これにより、2024 年 8 月から各省庁が所管する国家資格等の手続において、マイナポータルにてオンラインでの申請が可能となっている。また、国際的な取り組みとしては、国際標準化団体である World Wide Web Consortium (W3C) が検証可能クレデンシャル (VC: Verification Credential) の標準化に取り組んでいる [2]。VC によって、個人が自身の属性情報 (年齢、資格など) を必要に応じて選択的に開示することが可能となる。このように、国内外問わず属性情報をデジタル化する取り組みが広まっているため、安全かつ信頼性の高い認証および属性管理の必要性が高まっている。

属性認証の実現には、VC や属性ベース認証、高機能暗号技術、公開鍵基盤 (PKI) を用いた電子証明書ベースの技術が存在する。これらの中でも、PKI ベースの手法は長期にわたり使用されてきた技術として多くの国や企業で採用されている点などから、現実的な属性認証の実現法であると考えられる。しかし、従来の PKI ベースの属性証明書発行方式では、属性情報を管理する機関が証明書発行機関にユーザの機微な属性情報を提供する必要があり、プライバシー上の懸念が生じる。そのため、証明書発行機関に対し属性情報を秘匿した状態で属性証明書を発行できる仕組みが求められている。石井ら [3] はこのようなリスクを解決するため、ユーザの属性情報を証明書発行機関に秘匿したまま属性証明書を発行するための新しいシーケンスを提案した。彼らの方式は、ブラインド署名やハッシュ木といった暗号技術を用いることで、属性証明書発行時のプライバシー保護を実現しようとするものである。彼らは OpenSSL3.4.0 による概念実証を行っており、実用性についての議論は行われてはいない。本稿では、彼らの提案したシーケンスのうち特にブラインド署名を用いた属性証明書発行シーケンスの安全性について、形式検証により明らかにすることを目的とする。

暗号プロトコルの安全性評価は、その設計が複雑になるにつれて、人間の手作業による検証だけでは見逃しなどにより不完全になるリスクがある。このような課題に対し、形式手法は厳密にプロトコルの安全性を証明する有効な手段として注目されている。形式手法を適用することで脆弱性を発見し、安全性評価におけるヒューマンエラーを防止することが可能となる。本稿では、形式手法に基づく暗号プロトコル自動検証ツールの一つである ProVerif [4] に着目する。ProVerif は様々な性質を柔軟に検証可能である汎用性から、TLS 1.3 [5]、FIDO [6] や Bluetooth [7] など、広く普及しているプロトコルの検証に利用され、その安全性評価に貢献している。特に、国内においては、奥田らが

中央銀行によるデジタル通貨の一手法 [8] を提案しており、その安全性評価を形式検証で行っている [9], [10]。

1.2 貢献

本稿では、石井らが提案した属性認証におけるブラインド署名を用いた属性証明書発行シーケンスについて、暗号プロトコルの自動検証ツール ProVerif を用いて初めての形式検証を行う。本稿の具体的な貢献は以下の通りである。

- (1) ブラインド署名について、ブラインド化された状態での署名検証とアンブラインドされた状態での署名検証のそれぞれに対応した形式化を与える。
- (2) 属性管理機関が証明書発行機関以外から属性証明書を発行されないこと、証明書発行機関が属性管理機関以外に対して属性証明書を発行しないこと (なりすまし攻撃耐性)、および悪意のある証明書発行機関に対して属性情報を明らかにしないこと (秘匿性) を安全性要件として形式化を与える。
- (3) 上記の安全性要件を検証し、すべての要件を満たしていることを示す。

2. 準備

2.1 属性認証モデル

属性証明書の発行および利用には、ユーザ、属性管理機関、証明書発行機関、そして検証者が関与する。林田ら [11] は RFC5755 [12] に準拠した属性証明書の仕様を想定した属性認証モデルを提案した。具体的には、ユーザは X.509 形式の公開鍵証明書を所持しており、属性証明書はこの公開鍵証明書に紐づけて発行および使用されると仮定する。各エンティティの役割をまとめると次のようになる。

- ユーザ

属性証明書の発行を依頼するエンティティ。証明書発行機関に対する属性証明書の発行依頼は、属性管理機関を通して行う。

- 属性管理機関 (Attribute Manager: AM)

ユーザの属性情報を管理するエンティティ。ユーザからの証明書発行依頼を受け取って証明書発行機関に渡し、証明書を受け取った後にユーザに属性証明書を渡す。

- 証明書発行機関 (Certificate Issuer: CI)

属性証明書を発行するエンティティ。証明書発行依頼を検証したのち自身の署名鍵を用いて証明書を発行する。

- 検証者

属性証明書を検証するエンティティ。ユーザの属性情

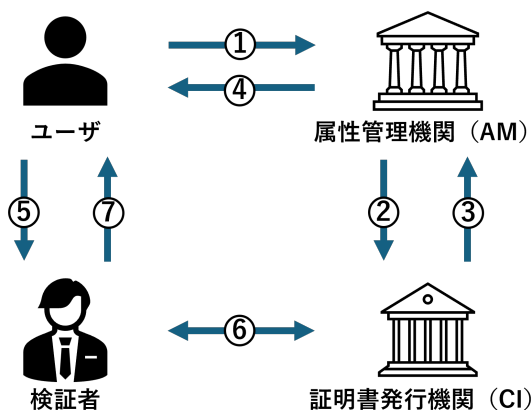


図 1 属性認証システムモデル

報の検証結果に応じてユーザへのサービス提供可否を判断する。

次に、属性証明書の発行と利用の流れを図 1 に示す。図中の各番号では以下の処理を行っている。

- ① ユーザによる AM に向けた属性証明書の発行依頼
- ② AM による CI に向けた属性証明書の発行依頼
- ③ CI による属性証明書の発行、AM へ送付
- ④ AM からユーザへ属性証明書の送付
- ⑤ ユーザから検証者へ属性認証依頼
- ⑥ 検証者による属性証明書と属性情報の検証
- ⑦ 検証者からユーザへサービス提供

また、機微な属性情報を取り扱うことを考慮し、以下の匿名性担保が重要となる。

- **証明書発行時の属性匿名性：**
証明書発行機関側にユーザの属性情報を直接渡さずに証明書を発行することが望ましい。これにより、証明書発行機関が情報銀行化してしまうリスクや情報漏洩リスクを低減する。
- **属性認証時の属性匿名性：**
ユーザが検証者に対して、認証に必要な属性情報だけを開示し、必要以上の属性情報を開示しないことが望ましい。
- **証明書失効確認時のユーザ匿名性：**
証明書の失効情報を確認する際に、第三者にユーザが利用しているサービスの情報が知られないようにすることが望ましい。

2.2 石井らの属性証明書発行・検証方式

石井らの方式では、2.1 節で述べた属性証明書の発行において、証明書発行時の属性匿名性を担保するものとなっている。属性情報の匿名性を実現するため、以下の 2 つの方式が提案されている。

- ブラインド署名を用いた属性証明書発行シーケンス：
証明書発行機関に対して属性情報を秘匿したまま属性証明書を発行する方式。
- ハッシュ木を用いた属性証明書発行・検証シーケンス：
属性情報の選択的開示を行う方式。

2.2.1 ブラインド署名を用いた属性証明書発行シーケンス

本稿の形式検証の対象となるのは、ブラインド署名を用いた属性証明書発行シーケンスである。ブラインド署名方式は RFC9474 [13] にて標準化されている RSASSA-PSS に基づく RSA ブラインド署名を用いることが想定されている。シーケンス全体の流れを図 2 に示す。1 つの属性証明書に署名対象のメッセージと署名値の組を 1 組しか含めることができないという技術的な制約を回避しつつ、属性情報そのものを証明書発行機関に秘匿したまま署名を得るために、以下の 2 つのセッションに分けて実行される。

第 1 セッション：属性管理機関は、属性情報を含まない証明書の基本情報（ユーザの公開鍵証明書のシリアル番号など）のみを含む仮の属性証明書発行依頼データを作成し、依頼データの電子署名とともに証明書発行機関に送付する。証明書発行機関は依頼データが正当であることを検証した上で、仮の属性証明書の基本情報（属性証明書発行者、シリアル番号などの情報）に対して署名を行い、仮の属性証明書として属性管理機関に送付する。

第 2 セッション：属性管理機関は属性証明書の基本情報と属性情報をまとめてブラインド化し、ブラインド化された申請情報を作成してブラインド化された申請情報の電子署名とともに証明書発行機関に送付する。証明書発行機関は、申請情報を検証したのち、ブラインド化された申請情報に対してブラインド署名を行い、ブラインド署名値を属性管理機関に返す。属性管理機関は受け取ったブラインド署名値をアンブラインドすることで元の申請情報の署名値を取得する。この署名値を、第 1 セッションで得られた仮の属性証明書に属性情報を付与したものに紐付け、最終的な属性証明書とすることで、属性情報が証明書発行機関に秘匿された状態で属性証明書が発行される。

また、石井らの方式では、通常の電子署名に用いる署名方式（RSASSA-PSS）に基づいてブラインド署名を構成しており、証明書発行機関が第 1 セッションで使用する通常

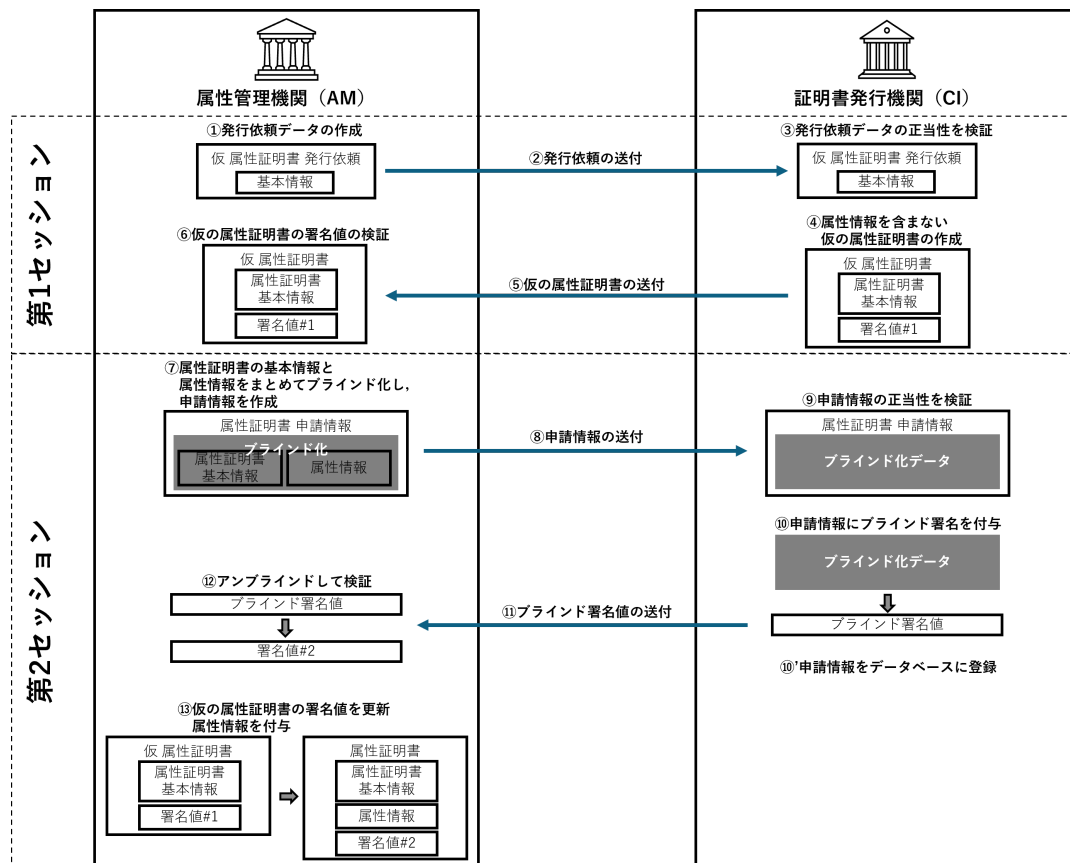


図 2 ブラインド署名を用いた属性証明書発行シーケンス

の電子署名と、第 2 セッションで使用するブラインド署名の両方に同一の鍵ペアを用いる。

3. ProVerif による形式化

本稿は、ProVerif [4] を用いて形式検証を行う。ProVerif は、Dolev-Yao [14] モデルに基づいて、メッセージや暗号プリミティブを記号に理想化して検証する暗号プロトコルの安全性自動検証ツールである。

3.1 構成要素の形式化

本稿で形式検証を行うシーケンスには、属性管理機関の公開鍵が正当であることを証明する公開鍵証明書が必要となる。本シーケンスでは、信頼できる第三者である ID プロバイダが、PKI に基づき公開鍵証明書を発行するものとしている。したがって、公開鍵証明書とその発行についても形式化の必要がある。形式化で用いる主要素について、以下に説明する。

3.1.1 基本的な情報の種類

- 通信路 `channel`：参加者間でメッセージをやり取りするための経路を表す。属性管理機関と証明書発行機関の間の通信路は公開通信路を仮定する。
- 公開鍵 `pkey` と秘密鍵 `skey`：暗号化や署名に用いられる、互に関連付けられた鍵のペアを表す。

- 乱数 `coins`：ブラインド署名において、メッセージをブラインド化するための値として使われる。
- メッセージの種類 `message_type`：プロトコル内でやり取りされるメッセージの種類を、発行依頼 `request`、証明書 `certificate`、申請情報 `application` といった定数として扱う。メッセージとメッセージの種類を組にして扱うことによって、メッセージの種類を区別する。

3.1.2 構成要素の暗号部品

- 公開鍵生成：秘密鍵から対応する公開鍵を導出する関数。公開鍵証明書発行用に用いられる関数 `pk_cert` と電子署名・ブラインド署名用に用いられる関数 `pk_sign` とでそれぞれ異なる。
- 公開鍵証明書の発行と検証：秘密鍵を用いてある公開鍵の証明書を生成する関数 `gen_cert` と、公開鍵証明書を検証する関数 `cert_ver`。公開鍵証明書がある秘密鍵によって生成されたとき、その秘密鍵に対応する公開鍵と証明対象の公開鍵は検証に受理される。
- 電子署名とその検証：秘密鍵を用いてメッセージに署名を施す関数 `sign` と、秘密鍵に対応する公開鍵を用いてメッセージと署名を検証する関数 `ver`。署名がある秘密鍵によって生成されたとき、その秘密鍵に対応す

る公開鍵と署名元のメッセージは検証に受理される。

- ブラインド署名とその検証
 - － ブラインド化：メッセージを乱数によってブラインド化する関数 `blind`.
 - － ブラインド署名：ブラインド化されたメッセージに対して、秘密鍵を用いて署名を施す関数 `blind.sign`.
 - － アンブラインド化：ブラインド化された署名値を、ブラインド化した際に使った乱数を用いてアンブラインド化する関数 `unblind`.
 - － ブラインド署名の検証：秘密鍵に対応する公開鍵を用いてアンブラインド化されたブラインド署名とメッセージを検証する関数 `blind.ver`. 秘密鍵に対応する公開鍵とブラインド署名と署名元のブラインド化されたメッセージは検証に受理される。また、秘密鍵に対応する公開鍵とアンブラインド化されたブラインド署名と署名元のブラインド化される前のメッセージは検証に受理される。本稿では、以下のよう

```
1 fun blind_ver(pkey, bitstring, bitstring):  
    bool  
2 reduc  
3     forall sk:skey, blinded_msg:bitstring;  
        blind_ver(pk_sign(sk), blinded_msg,  
        blind_sign(sk, blinded_msg)) = true  
4     otherwise forall sk:skey, msg:bitstring,  
        r:coins; blind_ver(pk_sign(sk), msg,  
        unblind(blind_sign(sk, blind(msg, r  
        )), r)) = true.
```

3.2 属性証明書発行シーケンスの形式化

石井らのブラインド署名を用いた属性証明書発行シーケンスの一連の手順を形式化する。我々の形式化では、エンティティとして属性管理機関と証明書発行機関だけを含め、本来必要となるユーザが属性管理機関に証明書の発行を依頼する手続きは省略している。その代わりに、発行依頼に含まれる基本情報や発行対象となる属性情報は、属性管理機関がシーケンスごとに生成する形で形式化している。また、シーケンス開始前に信頼できる ID プロバイダが、属性管理機関の公開鍵が正しいものであることを示す公開鍵証明書を発行する処理を加える。さらに、手順を単純化するため、ユーザが属性証明書を利用するときに検証者と証明書発行機関の間で必要となる、属性証明書の認証データを保存するなどの付随的な処理は、形式化の範囲から除いている。

発行依頼の基本情報と、それに基づいて発行される仮の属性証明書には、ホルダー領域に共通の公開鍵証明書のシリアル番号が含まれている。本稿では、属性管理機関が発

行依頼に対応する仮の属性証明書であるかを区別できるようにするため、発行依頼と仮の属性証明書それぞれの基本情報とホルダー情報を分けて形式化を行う。

3.2.1 シーケンス全体の流れ

属性管理機関と証明書発行機関の間でブラインド署名を用いて属性証明書を発行する全体的な流れを図 3 に示す。以下、各段階での具体的な処理について詳しく説明する。

- ① ID プロバイダが、属性管理機関の公開鍵証明書を発行する。
- ② 属性管理機関は、基本情報、ホルダー情報、およびそれに対する署名を仮の属性証明書発行依頼データとして作成する。
- ③ 属性管理機関は、発行依頼データと自身の公開鍵および公開鍵証明書を証明書発行機関に送付する。
- ④ 証明書発行機関は、属性管理機関から受け取った公開鍵が正しいかどうかを、公開鍵証明書を使って検証する。
- ⑤ 証明書発行機関は、属性管理機関の公開鍵の認証が成功したら、属性管理機関からの仮の属性証明書発行依頼に含まれる署名を検証する。
- ⑥ 証明書発行機関は、属性証明書の基本情報、ホルダー情報、およびそれに対する署名を含む仮の属性証明書を作成する。
- ⑦ 証明書発行機関は、仮の属性証明書を属性管理機関に送付する。
- ⑧ 属性管理機関は、受け取った仮の属性証明書に含まれる署名を検証する。
- ⑨ 属性管理機関は、属性証明書の基本情報、ホルダー情報と属性情報をまとめてブラインド化し、それに対する署名を含む属性証明書の申請情報を作成する。
- ⑩ 属性管理機関は、申請情報を証明書発行機関に送付する。
- ⑪ 証明書発行機関は、受け取った申請情報の署名を検証する。
- ⑫ 証明書発行機関は、申請情報の検証が成功すると、ブラインド化された申請情報に対してブラインド署名を行う。
- ⑬ 証明書発行機関は、ブラインド署名値を属性管理機関に送付する。
- ⑭ 属性管理機関は、受け取ったブラインド署名値にアン

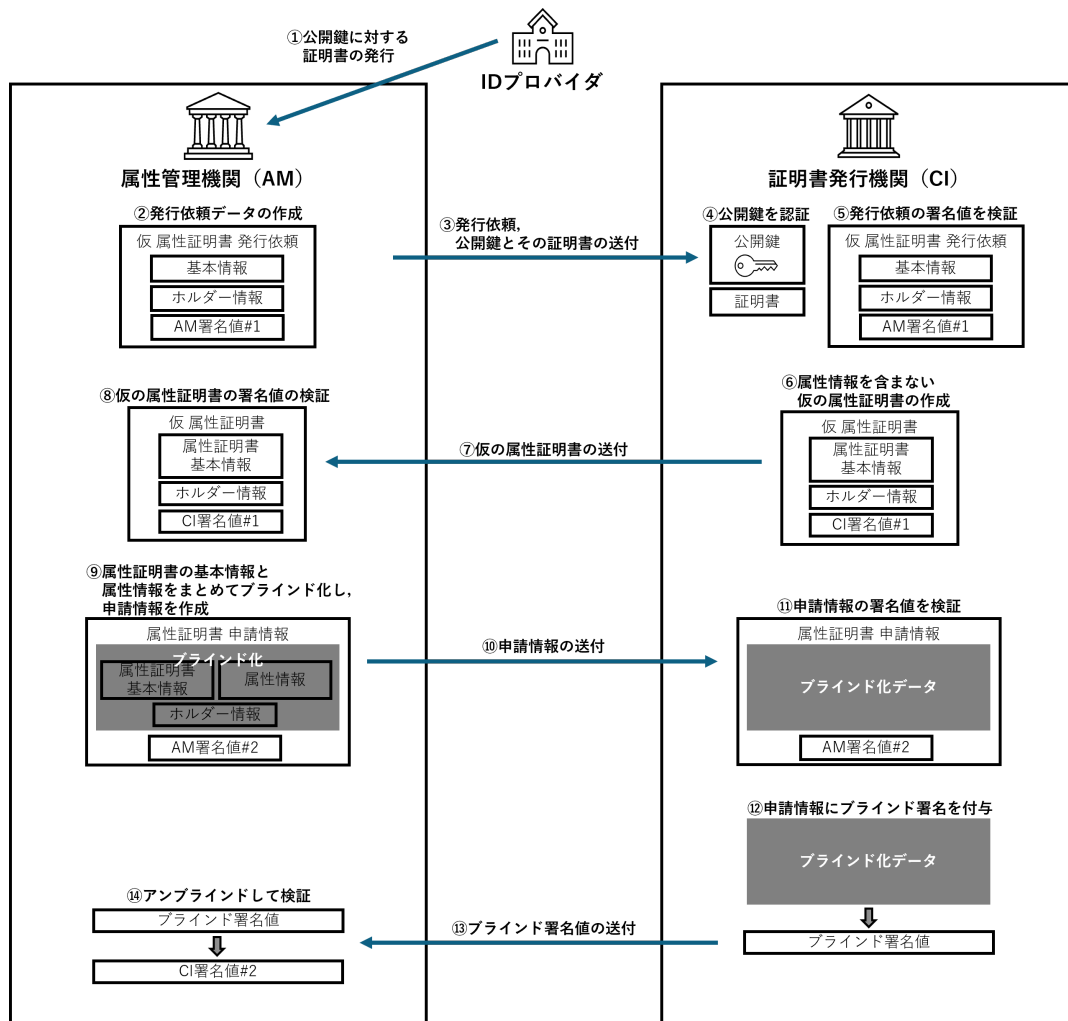


図 3 形式化したシーケンス全体

ブラインド化し、最終的な署名値を生成する。この署名値は、ブラインド化される前の元の属性証明書の基本情報、ホルダー情報と属性情報の組と証明書発行機関の公開鍵で検証する。

また、これらの手順は、2つの属性管理機関と1つの証明書発行機関の間で、複数プロセスが並行実行されるように形式化している。なお、2つの属性管理機関はそれぞれ固有の公開鍵・秘密鍵・公開鍵証明書を有する異なるサブプロセスを具体的に記述した。

3.2.2 悪意のある証明書発行機関を考慮する場合

本稿では、証明書発行機関が保有する機密情報を攻撃者に漏洩した場合（悪意のある証明書発行機関）を考慮する。証明書発行機関が保有する機密情報は自身の秘密鍵であるため、プロセスの開始時に、証明書発行機関の秘密鍵を公開されている通信路に送出することで攻撃者が利用可能となり、悪意のある証明書発行機関として動作することを表現する。

3.3 安全性要件の形式化

本稿では、以下の2つの安全性要件を形式化する。

- (1) **なりすまし攻撃耐性**：攻撃者が証明書発行機関に対して属性管理機関になりすますこと、または属性管理機関に対して証明書発行機関になりすますことができるか。
- (2) **秘匿性**：中間攻撃者や悪意のある証明書発行機関が、ユーザの属性情報に不正にアクセスできるか。

なりすまし攻撃耐性は、認証が正しく機能する条件を記述することで形式化する。

- **仮の属性証明書発行依頼における証明書発行機関側の形式化**：証明書発行機関が、仮の属性証明書発行依頼における基本情報の署名検証に成功した場合、その署名が2つの属性管理機関のどちらかによって生成されたものであるか。本稿では、2つの属性管理機関それぞれの仮の属性証明書発行依頼における署名を送信するイベント `send_sig1.am1`, `send_sig1.am2` と証明書発行機関が仮の属性証明書発行依頼における署名検証

に成功するイベント `ver_sig1_am_success` を定義し、以下のように形式化を行った。

```
1 query req_basic:bitstring;
2   event(ver_sig1_am_success(req_basic)) ==>
3   (event(send_sig1_am1(req_basic)) ||
4   event(send_sig1_am2(req_basic))).
```

- **属性証明書発行依頼における証明書発行機関側の形式化**：証明書発行機関が、属性証明書発行依頼における申請情報の署名検証に成功した場合、その署名が2つの属性管理機関のどちらかによって生成されたものであり、かつその前に仮の属性証明書発行依頼における基本情報の署名が、属性証明書発行依頼における申請情報の署名の送信元と同一の属性管理機関によって生成されているか。本稿では、2つの属性管理機関それぞれの仮の属性証明書発行依頼における署名を送信するイベント `send_sig1_am1`, `send_sig1_am2` と2つの属性管理機関それぞれの属性証明書発行依頼における署名を送信するイベント `send_sig2_am1`, `send_sig2_am2` と証明書発行機関が属性証明書発行依頼における署名検証に成功するイベント `ver_sig2_am_success` を定義し、以下のように形式化を行った。

```
1 query req_basic:bitstring, attribute:
   bitstring, cert_basic:bitstring, holder:
   bitstring, random:coins;
2   event(ver_sig2_am_success((application,
   blind((cert_basic, holder, attribute),
   random)))) ==>
3   ((event(send_sig2_am1((application, blind
   ((cert_basic, holder, attribute),
   random)))) ==>
4   event(send_sig1_am1((request, req_basic,
   holder)))) ||
5   (event(send_sig2_am2((application, blind
   ((cert_basic, holder, attribute),
   random)))) ==>
6   event(send_sig1_am2((request, req_basic,
   holder))))).
```

- **仮の属性証明書発行依頼における属性管理機関側の形式化**：属性管理機関が、仮の属性証明書発行依頼における属性証明書の基本情報の署名検証に成功した場合、その署名が証明書発行機関によって生成されたものであるか。本稿では、証明書発行機関が仮の属性証明書発行依頼における署名を送信するイベント `send_sig1_ci` と属性管理機関が仮の属性証明書発行依頼における署名検証に成功するイベント `ver_sig1_ci_success` を定義し、以下のように形式化を行った。

```
1 query cert_basic:bitstring;
2   event(ver_sig1_ci_success(cert_basic))
   ==>
3   event(send_sig1_ci(cert_basic)).
```

- **属性証明書発行依頼における属性管理機関側の形式化**：属性管理機関が、属性証明書発行依頼におけるアンブラインド化された申請情報の署名検証に成功した場合、対応するブラインド署名が証明書発行機関によって生成されたものであり、かつその前に仮の属性証明書発行依頼における属性証明書の基本情報の署名が証明書発行機関によって生成されているか。本稿では、証明書発行機関が仮の属性証明書発行依頼における署名を送信するイベント `send_sig1_ci` と証明書発行機関が属性証明書発行依頼における署名を送信するイベント `send_sig2_ci` と属性管理機関が属性証明書発行依頼における署名検証に成功するイベント `ver_sig2_ci_success` を定義し、以下のように形式化を行った。

```
1 query cert_basic:bitstring, attribute:
   bitstring, holder:bitstring, random:
   coins;
2   event(ver_sig2_ci_success((cert_basic,
   holder, attribute))) ==>
3   (event(send_sig2_ci(blind((cert_basic,
   holder, attribute), random))) ==>
4   event(send_sig1_ci((certificate,
   cert_basic, holder)))).
```

これらすべてのなりすまし攻撃耐性の要件が真である場合、要件に反するようななりすまし攻撃が起きないことを保証できる。

秘匿性の形式化では、以下の2つのシナリオを考慮する。

- **中間攻撃者に対する形式化**：中間攻撃者が通信経路上の情報から属性情報 `attribute_info` を取得できるか。本稿では、以下のように形式化を行った。

```
1 query secret attribute_info.
```

- **悪意のある証明書発行機関に対する形式化**：悪意のある証明書発行機関が、自身の秘密情報とシーケンス中に得られる情報から属性情報を取得できるか。形式化は上記のシナリオと同様である。

4. 検証結果

なりすまし攻撃耐性についての検証結果は以下の通りで

ある。

- 属性管理機関と証明書発行機関それぞれにおいて署名検証に成功した場合、その署名が他方によって生成されたものであると検証できたため、なりすまし攻撃耐性を有している。

秘匿性の検証結果については以下の通りである。

- 外部の中間攻撃者（証明書発行機関が攻撃者と結託しない場合）は属性情報に到達不可能。
- 悪意のある証明書発行機関（証明書発行機関の秘密鍵が漏洩した場合）は属性情報に到達不可能。

5. まとめ・今後の課題

5.1 まとめ

本稿では、石井らが提案したブラインド署名を用いた属性証明書発行シーケンスについて、ProVerif による形式検証を行った。結果として、彼らの方式はなりすまし攻撃耐性および証明書発行機関に対する属性情報の秘匿性をすべて満たすことを明らかにした。本稿における検証結果は、属性情報を秘匿したまま属性証明書を発行するという石井らの方式の安全性を形式的に裏付けるものである。

5.2 今後の課題

本稿では、ブラインド署名を用いたシーケンスを対象としたが、石井らはハッシュ木を用いた属性証明書発行・検証シーケンスについても提案している。今後の課題として、ハッシュ木を用いたシーケンスについても同様に形式検証を行い、その安全性を評価することが挙げられる。また、現実的な運用を考慮した課題として、属性証明書の利用、失効や更新といった属性認証システム全体を含めたモデルでの検証が重要となると考えられる。特に、これらの処理がプロトコル全体の安全性、特に秘匿性に与える影響を詳細に分析することが今後の研究の課題である。

参考文献

- [1] デジタル庁：国家資格等のオンライン・デジタル化，<https://www.digital.go.jp/policies/government-certification>.
- [2] World Wide Web Consortium: Verifiable Credentials Data Model v2.0, <https://www.w3.org/TR/vc-data-model/>.
- [3] 石井 龍，林田淳一郎，永井達也，野村健太，齋藤恆和，高田雄太，神薗雅紀：証明書発行者に対する匿名性をもつ属性証明書発行シーケンスおよび OpenSSL3.4.0 による概念実証，ICSS 研究会 (2025).
- [4] : ProVerif, <https://bblanche.gitlabpages.inria.fr/proverif/>.
- [5] Bhargavan, K., Blanchet, B. and Kobeissi, N.: Verified models and reference implementations for the TLS 1.3 standard candidate, *IEEE Symposium on Security and*

Privacy (SP), pp. 483–502 (2017).

- [6] Feng, H., Li, H., Pan, X., Zhao, Z. and Cactilab, T.: A Formal Analysis of the FIDO UAF Protocol, *Network and Distributed System Security Symposium (NDSS)* (2021).
- [7] Wu, J., Traynor, P., Xu, D., Tian, D. J. and Bianchi, A.: Finding Traceability Attacks in the Bluetooth Low Energy Specification and Its Implementations, *USENIX Security Symposium*, pp. 4499–4516 (2024).
- [8] 田村裕子，阿部正幸，奥田哲矢，津川天祐，宮澤俊之，山村和輝，赤羽喜治，田口智貴，平栗勇人，増田博人，山田健斗：「電子現金」に関する技術面からの一考察，2025 年 暗号と情報セキュリティシンポジウム (SCIS2025) (2025).
- [9] 奥田哲矢，荒井研一，齋藤恆和，千田浩司，中林美郷，山村和輝，宮澤俊之，阿部正幸：トークン型電子現金方式の形式検証手法に関する初期検討，研究報告コンピュータセキュリティ (CSEC)，Vol. 2022-CSEC-99, No. 24 (2022).
- [10] 奥田哲矢，荒井研一，齋藤恆和，千田浩司，中林美郷，山村和輝，宮澤俊之，阿部正幸：トークン型電子現金方式の二重使用検知およびプライバシーに関する形式検証の考察，研究報告コンピュータセキュリティ (CSEC)，Vol. 2023-CSEC-100, No. 66 (2023).
- [11] 林田淳一郎，石井 龍，永井達也，野村健太，齋藤恆和，高田雄太，神薗雅紀：属性管理と証明書発行の分担環境における PKI ベースの属性認証サービスモデルの提案とその実現可能性の検討，ICSS 研究会 (2025).
- [12] IETF Datatracker: RFC5755: An Internet Attribute Certificate Profile for Authorization, <https://datatracker.ietf.org/doc/rfc5755/> (2010).
- [13] IETF Datatracker: RFC9474: RSA Blind Signatures, <https://datatracker.ietf.org/doc/rfc9474/> (2023).
- [14] Dolev, D. and Yao, A.: On the security of public key protocols, *IEEE Transactions on Information Theory*, Vol. 29, No. 2, pp. 198–208 (2003).