属性間関係性を用いた属性認証における失効遅延削減方式

柿崎淑郎[†1] 辻 秀一[†2]

属性認証は被認証者の属性の正当性と有効性を確認することである．ある条件を満たしている場合にのみ付加される属性や，属性を証明する機関が様々であったり，属性は相互に複雑に関係し合ったりしている．属性を証明する電子証明書として，属性証明書がある．しかし，属性証明書は所有者を示すことはできるが，属性間の関係性を示すことができない．そのため，属性間の関係性をあらかじめ知らなくては検証できないことや，関係性のある属性証明書の失効に遅延が発生するといった問題点があった．本論文では，属性証明書が属性証明書間の関係性を示すことができるようにすることで，属性証明書の失効遅延を削減する方式を提案する．属性証明書の holder 領域を利用し，関係性のある属性証明書にバインドを行う．これにより，証明書検証者は属性間の関係性をあらかじめ知ることなく，属性証明書の情報から関係性のあるすべての属性証明書の有効性を検証することができる．また，属性証明書の失効遅延中であっても，失効遅延の影響を受けることなく属性証明書の有効性を検証できる．

A Method for Reducing Revocation Delay on a Relationship between Attributes in Attribute Authentication

Yoshio Kakizaki[†1] and Hidekazu Tsuji[†2]

Attribute authentication is to confirm validity of authenticatee's attribute. Attributes relate mutually and complexly; an attribute is issued only when the conditions are satisfied, and various authorities prove attributes. Attribute certificate as an electronic certificate proves attribute. However, attribute certificate cannot show the relationship although it can show the holder. Therefore, there were problems; it is impossible for verifier to verify the validity if it does not know the relationship between attributes beforehand, and revocation of attribute certificate with relationship is delayed. In this paper, we propose a foolproof method of verification process in attribute authentication. Attribute certificate is bound to attribute certificates that have the relationship by using the holder field of attribute certificate. Our method makes it possible that verifier can verify the validity of all attribute certificates with the relationship by using the holder field even if it cannot know the relationship between attributes beforehand. Moreover, verifier can verify the validity of attribute certificate without the influence of revocation delay although the revocation is being delayed.

1. はじめに

認証には大きく分けて，本人認証と属性認証がある．本人認証は被認証者が誰であるかを確認することであるのに対し，属性認証は被認証者が有する権限や属性を確認することである．属性認証によって，それに基づくアクセス制御やサービスの提供ができる[6),8)]．そのため，厳密な本人確認が必要でない場合においては，プライバシ保護を目的に利用される場合もある[1),12)]．

属性や権限は多種多様であり，相互に複雑に関係し合っている．ある条件が満たされないと生じない属性やある属性から生じる属性等もある．そのため，相互に複雑に関係し合う各属性間の関係性を確認することは，属性を利用・管理するうえで大変重要である．本論文では「属性と属性が関わり合っている状態とその性質」を「属性間の関係性」と呼ぶこととする．

属性証明書[2)]は証明書所有者の権限や属性を示す電子証明書である．属性証明書は証明書所有者の本人性を示す情報を持たないので，公開鍵証明書等へバインドすることで，証明書所有者の本人性を示す．しかしながら，属性証明書は本人性を示すことはできるが，他の属性証明書との関係性を示す方法は用意されていない．そのため，属性間の関係性を確認する手段や属性間の関係性を示す方法が必要であった．また，複数

†1 東海大学連合大学院理工学研究科
Graduate School of Science and Technology, Tokai University Unified Graduate School
†2 東海大学情報理工学部
School of Information Science and Technology, Tokai University

の異なる属性認証局によって発行された関係性を持った属性証明書の失効には，失効遅延が発生する問題があった．

本論文では属性間の関係性を示すことができる属性認証方式[4] を用いて，相互に複雑に関係し合っている属性証明書の失効遅延を削減する方式を提案する．従来方式は属性証明書が属性証明書間の関係性を示すことはできなかったが，本提案方式ではそれを可能とする．証明書検証者は検証しようとする属性証明書に加え，その属性証明書に関係するすべての属性証明書を検証できる．証明書検証者は属性証明書の情報を用いて，関係するすべての属性証明書を検証することができるので，関係性を持った属性証明書の失効遅延が発生している状態においても，その状態を検証することができ，即時に無効化することができる．

2. 関 連 研 究

2.1 属 性 認 証

属性認証は被認証者がどのような権限や属性を持っているかや，その権限や属性の正当性を認証することである．属性証明書（Attribute Certificate; AC)[2] は権限や属性を証明する電子証明書であり，**表 1** に示す領域で構成される．属性証明書は公開鍵証明書（Public Key Certificate; PKC)[3] と同じ X.509 証明書だが，属性証明書は権限や属性の情報を持つ反面，本人性を示す情報を持たないという特徴がある．そのため，実際に属性証明書を利用する場合は，本人性を示す情報を持つ公開鍵証明書による本人認証と組み合わせる．これにより，公開鍵証明書による本人認証と属性証明書による属性認証が実現される．

属性証明書による属性認証は属性証明書の発行と失効，属性証明書の有効性検証の 3 つの手順により行われる．属性証明書の発行では，被認証者の属性を記載し，被認証者の公開鍵証明書等にバインドした属性証明書に，属性認証局がディジタル署名を行い，発行する．属性証明書の失効では，属性の変更や消失等の理由によって，発行されている属性証明書の効力を無効化するために，属性認証局が発行している属性証明書失効リスト（Attribute Certificate Revocation List; ACRL）に失効させる属性証明書を記載することで行われる．属性証明書の有効性検証では，属性証明書が有効期限内であることと，属性証明書が失効していないことを ACRL によって確認する．

表 1 属性証明書のプロファイル
Table 1 Profile of attribute certificate standard fields.

領域名	説明
version	v2
holder	証明書所有者を識別するための情報
issuer	発行した属性認証局の名前
signature	発行者の署名アルゴリズム
serialNumber	証明書を識別するための番号
attrCertValidityPeriod	有効期限
attributes	属性情報
issuerUniqueID	issuer を識別するための識別子
extensions	証明書の拡張領域

属性情報に関する研究として千葉らの研究[13] がある．千葉らは個人属性を安全に交換・管理する情報化基盤として，属性情報プロバイダを提案している．属性情報プロバイダに属性情報を登録することで，属性情報の実用性や信頼性を向上させている．しかし，属性情報を集中管理するため，属性プロバイダの運用や分散化された情報の管理責任の明確化等の検討課題が残されている．

先に述べたように，属性証明書は権限や属性の情報を持つ反面，本人性を示す情報を持たないため，属性証明書だけでは証明書所有者の本人性を証明することができない．証明書所有者の本人性を証明するために，属性証明書の holder 領域を用い，証明書所有者の公開鍵証明書等にバインドすることで，属性とその所有者を結び付けている．属性証明書の holder 領域には baseCertificateID，entityName，objectDigestInfo の 3 つのオプションがある．**表 2** に holder 領域の 3 つのオプションを示す．

baseCertificateID は公開鍵証明書の serialNumber と issuer を示すことで，属性証明書所有者を特定する．entityName は公開鍵証明書の subject または subjectAltName を示すことで，属性証明書所有者を特定する．objectDigestInfo は対象となるオブジェクトのハッシュ値を示すことで，属性証明書所有者を特定する．このように，属性証明書の holder 領域は属性証明書所有者の本人性を確認するための情報を示すことができる．

2.2 属性間の関係性を確認する従来方式

各属性情報を含んだ属性証明書は，**図 1** に示すように，属性証明書の holder 領域によって，属性証明書所有者の公開鍵証明書等にバインドされる．図 1 中の AC1 と AC3 は AC2 と依存関係にあるが，AC2 は PKC にバインドされているだけで，属性証明書はその関係性を提示することができない．属性証明書を利用した従来の属性認証では，提示された属性証明書の有効性と正当性およびバインド先である公開鍵証明書を検証することはできるが，属性間の関係性を検証することができないという問題点がある．

表 2 holder 領域のオプション
Table 2 Profile of the holder field options.

オプション名	説明
baseCertificateID	所有者の PKC の serialNumber と issuer
entityName	所有者の PKC の subject または subjectAltName
objectDigestInfo	対象物のハッシュ値

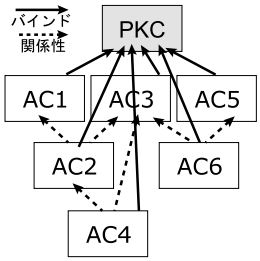

図 1 公開鍵証明書と属性証明書をバインドする従来方式
Fig. 1 Naive method of binding PKC and AC.

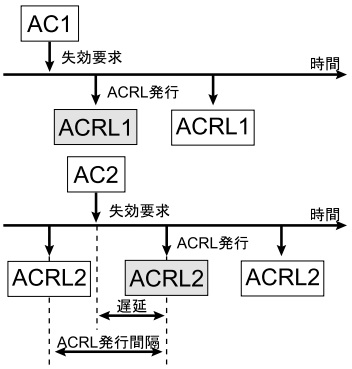

図 2 従来方式における失効遅延の問題
Fig. 2 Issue of revocation delay in naive method.

属性認証における属性間の関係性を確認する従来方式として，以下の方法が用いられる．

(1) 関係性のある属性を 1 枚の属性証明書に記載する方法[15)]

(2) 属性間の関係性を把握する方法

関係性のある属性を 1 枚の属性証明書に記載する方法は，関係性のある複数の属性が 1 枚の属性証明書に記載されるため，関係性のある属性を確認することができる．この方法の場合，属性証明書内で最も有効期限が短い属性にすべての属性が依存するため，属性の変更や失効にともなう属性証明書の再発行頻度が高くなる．また，関係性のあるすべての属性を単一の属性認証局が証明する必要性があるため，どの属性認証局が属性証明書を発行するのかという問題がある．

あらかじめ属性間の関係性を把握しておくことで，属性証明書の失効や再発行にともなう整合性をとることができる．この方法を属性間の関係性を把握する方法とする．属性認証局は発行した属性証明書に関する属性間の関係性を把握している．また，属性認証局から権限委譲を行われたサービスプロバイダや属性プロバイダ等も知りうる可能性がある．図 1 中の AC2 を発行した属性認証局は AC2 の発行に際して，AC1 と AC3 を検証しており，その関係性を把握している．そのため，AC2 を発行した属性認証局は AC1 と AC3 を定期的にチェックすることで，AC2 を失効させることができる．しかしその反面，属性証明書の発行枚数が多くなったり，属性間の関係性が複雑化したりした場合，属性間の関係性を把握している機関の処理負荷が増大する．

この方法では属性証明書が属性証明書間の関係性を提示できないために，**図 2** に示すような失効遅延の問題がある．AC2 は AC1 に依存しており，AC2 を発行する属性認証局は AC1 の状態を ACRL1 で確認している．図 2 では，まず AC1 の失効処理が始まり，AC1 が ACRL1 に記載される．AC1 が ACRL1 に記載されたことを AC2 を発行する属性認証局が確認したら，AC2 を発行する属性認証局は AC2 の失効処理を始める．この時点で，AC2 はまだ ACRL2 に記載されていないので，失効していない．そのため，AC1 が失効してから ACRL2 に AC2 が記載されるまでの遅延が生じる．この失効遅延の時間差を利用して，不正利用者は属性証明書を不正に使用することができる．ACRL の発行間隔を短くすれば，失効遅延を短くできるが，属性認証局の処理コストが増加する．逆に，発行間隔を長くすれば，属性認証局の処理コストが低減される反面，失効遅延が長くなる．これらはトレードオフの関係である．

一般に，属性認証局は発行した属性証明書とその発行に必要な属性証明書との関係性を把握しているが，発行に必要な属性証明書よりも上位の関係性は把握していない．把握している属性間の関係性以外は把握していないので，階層的関係性を持つ属性証明書の失効遅延時間は，上位において関係性がある属性証明書が失効した場合に，その階層が深くなるにつれて増大する．

3. 失効遅延削減方式

属性証明書には属性間の関係性を示す確実な方法がない．関係性のある属性を 1 枚の属性証明書に記載する方法は，関係性のあるすべての属性が 1 枚の属性証明書に含まれているため，属性間の関係性を検証することができるが，どの属性認証局が発行するのかという問題がある．属性証明書自身が属性間の関係性を提示できないことによって，属性間の関係性を把握する方法においては，誰もが自由に検証を行うことができず，また失効遅延の問題がある．

これらの問題を解決するために，属性証明書が属性間の関係性を提示することによって，属性証明書の失効遅延を削減する方式を提案する．これにより，属性証明書が提示する属性間の関係性を利用することで，誰もが自由に検証を行うことができる．また，検証時に関係性のあるすべての属性証明書を検証することができるので，失効遅延によらず，属性証明書を無効にすることができる．各属性証明書は関係性を提示できるので，関係性のある属性を 1 枚の属性証明書にする必要がなく，各属性は異なる属性認証局によって発行されることができる．

3.1 定　　義

ここで，本論文で用いる用語の定義を行う．

属性間の関係性は親子関係で示す．関係性のある属性が依存している上位の属性を親属性と呼び，親属性が含まれている属性証明書を親属性証明書と呼ぶ．同様に，関係性のある属性に依存している下位の属性を子属性と呼び，子属性が含まれている属性証明書を子属性証明書と呼ぶ．また，この親子関係は 2 つまたは複数の属性間の関係性を示すだけであって，属性の本質を示すものではない．

公開鍵証明書にバインドされる属性証明書を一次属性証明書と呼ぶ．そのため，一次属性証明書は子属性を含むことはなく，子属性証明書にはなりえない．

これらの関係を**図 3** に示す．

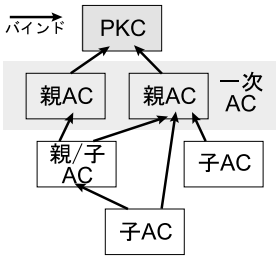

図 3 属性証明書とその関係性の定義
Fig. 3 Term definition of attribute certificates and its relationship.

3.2 アプローチ

従来方式は図 1 に示すように，各属性証明書がその所有者の公開鍵証明書にバインドされていた．図 1 の破線は属性間の関係性を示しているが，属性証明書自体は属性証明書間の関係性を示すことができなかった．そこで，本提案方式では図 3 のように，属性証明書が属性間の関係性を提示できる方法を考えた．通常の属性証明書の baseCertificateID は公開鍵証明書しかバインドすることができない．属性証明書の extensions 領域によって，属性証明書をバインドできるように拡張することはできるが，holder 領域は属性証明書に必ず必要な項目であり，holder 領域に所有者の公開鍵証明書をバインドする必要がある．そのため，図 3 のように属性間の関係性を提示できる属性証明書を目的として，baseCertificateID が属性証明書をバインドできるようにし，また holder 領域が複数の値を持つことができるように，独自拡張するアプローチを行った．holder 領域における属性証明書のバインドには，baseCertificateID と objectDigestInfo を用いる．

baseCertificateID を用いる方法

本来，baseCertificateID は公開鍵証明書しかバインドすることができない．そこで，本提案方式では属性証明書もバインドできるように独自拡張を行う．この方法は親属性証明書の serialNumber と issuer を指定することで，一意に親属性証明書を指し示すことができるため，最も結合強度の強いバインド方法である．

この方法は親属性証明書と子属性証明書を密にバインドし，子属性証明書から親属性証明書を一意に特定することができる．親属性証明書の serialNumber と issuer が分かれば，その親属性証明書を発行している属性認証局のリポジトリから，親属性証明書を取得することができる．そのため，属性証明書の有効性を検証しようとする検証者は，子属性証明書のバインドをたどることで，親属性証明書の有効性を検証することが可能である．

objectDigestInfo を用いる方法

この方法は親属性証明書のハッシュ値を指定することで，親属性証明書を指し示す方法である．ハッシュ値には衝突の可能性があり一意ではないため，属性証明書間の結合が 1 方向性となり，強度が強くはない．

ハッシュ値は 1 方向性関数で算出されるため，ハッシュ値から親属性証明書を特定することは困難である．子属性証明書の有効性検証時には，ハッシュ値から親属性証明書を特定することができないので，親属性証明書の提示が必要になる．また，ハッシュ値の衝突の

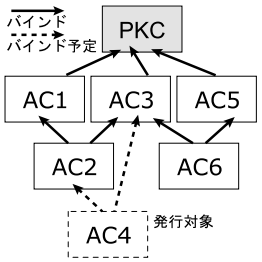

図 4 提案方式の処理方式のための例図
Fig. 4 An example of process for issuing, verification, revocation.

可能性があるが，その可能性はハッシュアルゴリズムの衝突困難性に依存する．

逆に，この方法はハッシュ値を用いるため，属性証明書のハッシュ値以外にも，属性証明書のディジタル署名やバイオメトリクス情報等，ハッシュ値が計算できるあらゆるオブジェクトにバインドできるという柔軟性がある．

3.3 属性証明書の処理方法

提案方式である属性間の関係性を考慮した属性認証方式で行われる，発行処理，有効性検証と無効化，失効処理の 3 つについて説明する．ここでは**図 4** を例として説明する．

発 行 処 理

一次属性証明書の発行は従来方式と変わらず，公開鍵証明書にバインドした属性証明書を属性認証局が発行する．つまり，AC1，AC3，AC5 を発行する場合は，従来と同様の発行処理手順である．

関係性を持った属性証明書の発行は以下の手順で行う．

(1) 発行しようとする子属性証明書に関係するすべての親属性証明書の有効性を検証する．
(2) 親属性証明書のバインドをたどり，再び同じ親属性証明書にたどり着いた場合，属性証明書間の関係性が循環参照になっているため，属性証明書の発行処理を中止する．
(3) 検証する親属性証明書がさらに他の属性証明書の子属性証明書である場合，再帰的に上位の親属性証明書の有効性を検証する．
(4) 検証する属性証明書が一次属性証明書の場合，検証すべきすべての一次属性証明書が同一の公開鍵証明書にバインドされていて，かつその公開鍵証明書の有効性を検証する．
(5) 以上のすべての手順が完了したら，発行しようとする子属性証明書をすべての親属性証明書にバインドし，発行する．

図 4 の AC4 を発行する際を例にあげて説明する．AC4 に関係する親属性証明書は AC2 と AC3 であるので，AC2 と AC3 の有効性を検証する．AC2 は AC1 と AC3 の子属性証明書であるので，AC2 の親属性証明書である AC1 と AC3 の有効性を検証する．AC1 と AC3 は一次属性証明書であるので，AC1 と AC3 がともに同じ PKC にバインドされていて，PKC が有効であることを検証する．以上のすべての手順が完了したら，AC4 の holder 領域に AC2 と AC3 を記載し，発行する．

発行された子属性証明書は holder 領域によって，親属性証明書にバインドされている．その親属性証明書はさらに上位の親属性証明書や公開鍵証明書にバインドされている．そのため，発行された子属性証明書から holder 領域の情報をたどることで，関係性のあるすべての属性証明書と公開鍵証明書を示すことができる．

有効性検証と無効化

関係性を持った属性証明書の有効性検証は以下の手順で行う．

(1) 検証しようとする属性証明書が有効期限内であり，失効していないことを検証する．
(2) 検証しようとする属性証明書の holder 領域から，親属性証明書の有効性を検証する．
(3) 親属性証明書がさらに他の属性証明書の子属性証明書である場合，再帰的に上位の親属性証明書の有効性を検証する．
(4) 検証しようとする属性証明書から親子関係をたどって到達できるすべての一次属性証明書が同一の公開鍵証明書にバインドされていることを確認する．
(5) 公開鍵証明書が有効期限内であり，失効していないことを検証し，属性証明書利用者がこの公開鍵証明書の所有者であることを確認する．
(6) 以上のすべての手順が完了したら，検証しようとする属性証明書は有効である．そうでなければ，検証しようとする属性証明書は無効である．

もし，検証対象の属性証明書よりも上位の親属性証明書が失効している場合，手順 (3) により，属性証明書の有効性が検証できなくなる．そのため，失効している親属性証明書よりも下位のすべての子属性証明書は，有効性が検証できなくなるため，ただちに無効となる．また，属性証明書の有効期限切れや再発行の場合においても，同様に有効性が検証できなくなる．

図 4 において，AC4 を検証対象の子属性証明書とし，AC1 が失効状態であるとして，有効性検証の例を示す．AC1 は失効状態であるが，AC2 はまだ失効

しておらず，図 2 に示すような失効遅延が発生している状態である．そのため，従来方式では AC2 の有効性が確認できるため，AC4 を使用できる状態である．提案方式では手順 (2) に従い，AC4 の holder 領域によって，親属性証明書である AC2 と AC3 の有効性を検証する．さらに，手順 (3) に従い，AC2 の holder 領域によって，親属性証明書である AC1 と AC3 の有効性を検証する．ここで，AC1 が失効状態であることが確認されるため，AC2 はただちに無効となる．AC2 が無効となるならば，AC2 を親属性証明書とする AC4 もただちに無効とすることができる．このように，本提案方式は失効遅延中であっても，検証対象の属性証明書をただちに無効にすることができる．

失 効 処 理

属性証明書の失効は従来方式と同様に行われる．すなわち，属性の変化や失効にともない，有効期限内の属性証明書は属性認証局によって失効される．失効は属性認証局が発行する ACRL によって行われる．

本提案方式では属性証明書が属性間の関係性を示すことができるため，親属性証明書の失効にともなう子属性証明書の失効処理は，即時に実行される必要がない．検証者は子属性証明書のバインドをたどり，親属性証明書の有効性を検証することができるからである．これにより，従来方式の問題点であった失効遅延は問題とならない．そのため，ACRL の発行間隔を長くすることが可能であり，属性認証局の処理コストが低減される．

4. 評　　価

4.1 適用例による評価

本提案方式の適用例を示して，評価する．本提案方式が効果的に作用する例として，ドライバ保険サービスをあげる．この適用例では，各属性を同一の属性認証局で管理することは事実上不可能である．ドライバ保険の加入条件は運転免許証を持っていることであり，1 年契約の更新制とする．保険等級は 1 年間の無事故で 1 等級増，事故を起こすたびに 3 等級減とし，等級が上がるほど保険料の割引率が高い．運転免許証は 3 年ごとに更新が必要とする．

4.1.1 証　明　書

本適用例で用いる公開鍵証明書と属性証明書を以下のように定義する．また，それぞれの関係性を**図 5** に示す．ここで，holder 領域には baseCertificateID を用いることとする．

- PKC：対象者の公開鍵証明書
- AC_DL：運転免許証の属性証明書

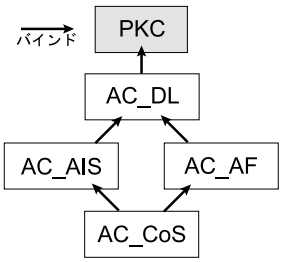

図 5 提案方式の適用例
Fig. 5 An application model of our method.

- AC_AF：無事故期間の属性証明書
- AC_AIS：ドライバ保険証の属性証明書
- AC_CoS：保険等級の属性証明書

各属性証明書を以下のように定義する．

AC_DL

- holder：PKC
- issuer：公安委員会
- attributes：普通自動車免許証
- validityPeriod
  - notBeforeTime: 20050712
  - notAfterTime: 20080831

AC_AF

- holder：AC_DL
- issuer：警察署
- attributes：無事故期間
- validityPeriod
  - notBeforeTime: 19990312

AC_AIS

- holder：AC_DL
- issuer：自動車保険会社
- attributes：ドライバ保険証
- validityPeriod
  - notBeforeTime: 20070401
  - notAfterTime: 20080331

AC_CoS

- holder：AC_AIS，AC_AF
- issuer：自動車保険会社
- attributes：保険等級 14 等級
- validityPeriod
  - notBeforeTime: 20070401
  - notAfterTime: 20080331

4.1.2 シ ナ リ オ

シナリオ 1：AC_CoS の失効と再発行

AC_CoS は AC_AIS の発行と同時に失効および再発行されるが，AC_AF の失効および再発行によってもまた，失効および再発行が行われる．つまり，AC_CoS

の再発行は AC_AIS および AC_AF の再発行によって行われる．

保険加入者がドライバ保険契約を更新し，AC_AIS が失効し，新しいドライバ保険証 AC_AIS' が発行された場合を考える．保険加入者は AC_CoS を自動車保険会社に提示し，割引サービスを利用しようとする．自動車保険会社は提示された AC_CoS の holder 領域から，親属性証明書である AC_AIS と AC_AF の有効性を検証するために，AC_AIS と AC_AF を発行する属性認証局のリポジトリから，それぞれの属性証明書を取得する．AC_AIS は失効しているため，その有効性を検証できず，AC_CoS は無効となる．保険加入者は AC_CoS を発行する属性認証局に新しい AC_CoS の発行を要求する．AC_CoS を発行する属性認証局は AC_AIS の再発行にともない，AC_AF を確認して 1 年間の無事故であれば，1 等級増して AC_CoS 発行する．

保険加入者が事故を起こし，AC_AF が失効した場合を考える．保険加入者は AC_CoS を自動車保険会社に提示し，割引サービスを利用しようとする．自動車保険会社は提示された AC_CoS の holder 領域から，親属性証明書である AC_AIS と AC_AF の有効性を検証するために，AC_AIS と AC_AF を発行する属性認証局のリポジトリから，それぞれの属性証明書を取得する．AC_AF は失効しているため，その有効性を検証できず，AC_CoS は無効となる．保険加入者は AC_CoS を発行する属性認証局に新しい AC_CoS の発行を要求する．AC_CoS を発行する属性認証局は AC_AF の再発行にともない，AC_CoS を 3 等級減じて発行する．

シナリオ 2：AC_DL の失効

更新のし忘れや違反累積等によって AC_DL が失効した場合を考える．保険加入者は AC_AIS を自動車保険会社に提示し，保険適用を受けようとする．自動車保険会社は提示された AC_AIS の holder 領域から，親属性証明書である AC_DL の有効性を検証するために，AC_DL を発行する属性認証局のリポジトリから属性証明書を取得する．AC_DL は失効しているため，その有効性を検証できず，AC_AIS は無効となる．

4.1.3 検証処理の妥当性

関係性を持った属性証明書が使われる一例として，ドライバ保険サービスをあげ，本提案方式を適用した．属性証明書は各々1 つの属性を含んでおり，各属性証明書は独立に利用することが可能である．また，本提案方式を適用したことにより，検証者は属性証明書のバインドをたどることで，関係性を持った属性証明書とその所有者の公開鍵証明書を検証することができる．これにより，親属性証明書が失効した場合，子属性証明書の有効性が検証できなくなるため，子属性証明書を即時に無効化できる．

このように，関係性を持った属性証明書が使われる事例に対して本提案方式を適用することで，親属性証明書の失効にともない，子属性証明書を即時に無効化することができる．また，それぞれの属性証明書が複数の属性認証局によって発行されていても，属性証明書のバインドをたどることで，属性証明書間の関係性を検証することができる．そのため，関係性を持った属性証明書が使われる属性認証に広く適用することができる．

4.2 安全性の評価

属性証明書は X.509 証明書であり，属性認証局によってディジタル署名されているため，属性証明書の捏造と改ざんについての安全性は従来方式と変わらない．

第三者が属性証明書を不正に入手して利用する場合を考える．baseCertificateID を使う方法は，バインドをたどることで公開鍵証明書を一意に特定することができるため，第三者が利用することはできない．objectDigestInfo を使う方法はバインドにハッシュ値を用いるので，もしも同じハッシュ値の公開鍵証明書や属性証明書があれば，所有者を偽って利用することが可能である．しかし，ハッシュ値を算出するためのハッシュアルゴリズムには，第 2 原像計算困難性，衝突困難性の特徴があるため，実用的な攻撃にはなりえない．

5. 考　　察

5.1 属性の関係について

属性の関係性は図 4 のような階層構造を持つことが望ましい．しかしながら，属性には明確な順位付けが存在していないので，同じ属性であっても，階層が異なる扱いをされる場合がある．そのため，階層構造ではなく，循環参照の状態になる可能性がある．本提案方式では発行処理の手順 (2) において，循環参照であった場合の属性証明書発行を行わないようにしている．

本提案方式は属性間の関係性を属性証明書が示すことができるようにすることで，証明書検証者が検証しようとする属性証明書とそれに関係するすべての属性証明書を検証できるようにする方式である．本提案方式上で循環参照が発生する場合，属性自体の関係性が循環参照の状態になっている．よって，循環参照の問

題は本提案方式の問題ではなく，属性自体の関係性を管理する際の問題であるといえる．

5.2 バインド方法の比較

関係性のある属性証明書をバインドするために，属性証明書の holder 領域を用いる．本提案方式では holder 領域に baseCertificateID と objectDigestInfo を用いることができる．

baseCertificateID を用いる方法は親属性証明書の serialNumber と issuer を指定することで，一意に親属性証明書を指し示すことができるため，最も結合強度の強いバインド方法である．子属性証明書から親属性証明書を一意に特定することができるので，属性証明書検証者は親属性証明書の有効性を検証することが可能である．

一方で，objectDigestInfo を用いる方法はハッシュ値でバインドを行う．ハッシュ値は 1 方向性関数で算出されるため，ハッシュ値から親属性証明書を特定することは困難である．そのため，子属性証明書の有効性検証時には親属性証明書の提示が必要になる．この方法はバインドにハッシュ値を用いるため，属性証明書のディジタル署名やバイオメトリクス情報等，ハッシュ値が計算できるあらゆるオブジェクトにバインドできるという柔軟性がある．

バインド先の属性証明書や公開鍵証明書が再発行された場合，baseCertificateID を用いる方法では属性証明書の再発行が必要となる．しかし，バインド先の属性証明書や公開鍵証明書が再発行されても，その公開鍵自体は変更されないので，objectDigestInfo を用いる方法では属性証明書の再発行が不要である．そのため，子属性証明書よりも親属性証明書の有効期限が短い場合に有効なバインド方法である．

一次属性証明書に該当する属性情報は年齢や性別等の個人に近い属性情報が多く，永年属性を含め，有効期限が長期の属性情報であることが一般的である．たとえば，年齢を示す属性証明書は生年月日を証明すればよく，生年月日は不変であるので，永年属性として有効期限を “never revoke” にできる．公開鍵証明書は通常数年程度の有効期限がある．このような状況下では，公開鍵証明書と属性証明書を baseCertificateID でバインドすると，公開鍵証明書の再発行にともない，属性証明書の再発行が必要となる．ここで，公開鍵のハッシュ値と属性証明書を objectDigestInfo でバインドすることで，公開鍵が変わらない限り，公開鍵証明書の再発行にともなう属性証明書の再発行は不要となる[14]．これにより，年齢を示す属性証明書を親属性証明書とする子属性証明書は，公開鍵証明書の再発行に影響されず，属性証明書の不必要な再発行が抑制される．

今枝らは文献 9) において，1 世代属性証明書と複数世代公開鍵証明書をバインドする際の問題について言及しているが，本提案方式はその影響が属性証明書間においても発生する．そのため，有効期限が短い親属性証明書へのバインドには objectDigestInfo を用い，有効期限が長い親属性証明書へのバインドには baseCertificateID を用いることで，属性証明書の不必要な再発行が抑制しつつ，効率的な属性証明書の運用が可能となる．

5.3 証明書検証コスト

本提案方式では 3.3 節の有効性検証に従い，証明書検証者は検証しようとする属性証明書からバインドをたどり，関係性のあるすべての親属性証明書と公開鍵証明書を検証する必要がある．そのため，従来方式である属性間の関係性を把握する方法に比べて，本提案方式における検証者の証明書検証コストは増加する．

図 1 の AC4 を検証する場合を例にあげる．従来方式である属性間の関係性を把握する方法では，AC4 を発行した属性認証局によって，AC4 と関係性にある AC2 と AC3 は定期的にチェックされている．そのため，検証者は AC4 とその所有者の PKC を検証するだけでよく，検証回数は 2 回となる．この方法では検証者の証明書検証コストは低いが，関係性にある属性証明書を定期的にチェックしている属性認証局の証明書検証コストは高い．

提案方式の場合は図 4 の AC4 を検証することになる．検証者は検証しようとする属性証明書 AC4 の有効性を検証するために，AC4 の親属性証明書である AC2 と AC3 の有効性を検証する必要がある．AC2 の有効性検証には AC1 と AC3 の有効性を検証する必要がある．さらに，AC1 と AC3 の有効性検証にはそれぞれの属性証明書所有者の公開鍵証明書を検証する必要がある．つまり，検証回数は 7 回となる．この方法では検証者の証明書検証コストは従来方式に比べ増加しているが，属性認証局の負担は軽減される．

提案方式における検証者の証明書検証コスト削減は今後の課題となる．

6. おわりに

本論文では相互に複雑に関係し合っている属性証明書の失効遅延問題を解決するべく，属性間関係性を用いた属性認証によって，失効遅延を削減する方式を提案した．従来方式では属性証明書間の関係性を示す方法がなく，属性証明書の有効性検証・失効処理に問題

があった．本提案方式は属性証明書の holder 領域を用い，属性証明書が属性証明書間の関係性を示すことができるようにしたことで，証明書検証者は検証しようとする属性証明書に加え，その属性証明書に関係するすべての属性証明書を検証できる．証明書検証者は属性証明書の情報を用いて，関係するすべての属性証明書を検証することができるので，関係性を持った属性証明書の失効遅延が発生している状態においても，その状態を検証することができ，即時に無効化することができる．これにより，失効処理の遅延による時間差を用いた不正利用を防止することができる．また，関係性を持った属性証明書が複数の異なる属性認証局から発行されていても適用できるため，本提案方式は様々な属性認証に広く適用することができる．一方で，従来方式に比べ，証明書検証コストは増加している．この削減については今後の検討課題となる．

参 考 文 献

1) Benjumea, V., Lopez, J., Montenegro, J.A. and Troya, J.M.: A First Approach to Provide Anonymity in Attribute Certificates, *PKC 2004*, Vol.2947 of LNCS, pp.402–415 (2004).
2) Farrell, S. and Housley, R.: An Internet Attribute Certificate Profile for Authorization, RFC3281 (2002).
3) Housley, R., Polk, W., Ford, W. and Solo, D.: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, RFC3280 (2002).
4) Kakizaki, Y. and Tsuji, H.: A New Method for Reducing the Revocation Delay in the Attribute Authentication, *ARES 2007*, pp.1175–1182 (2007).
5) Nakanishi, T. and Sugiyama, Y.: Anonymous Statistical Survey of Attributes, *ACISP2001*, Vol.2119 of LNCS, pp.460–473 (2001).
6) Oppliger, R., Pernul, G. and Strauss, C.: Using Attribute Certificates to Implement Role-based Authorization and Access Controls, *SIS 2000* (2000).
7) Park, J.S. and Sandhu, R.: Binding Identities and Attributes using Digitally Signed Certificates, *16th ACSAC*, pp.120–127 (2000).
8) Zhou, W. and Meinel, C.: Implement role based access control with attribute certificates, *ICACT2004*, Vol.1, pp.536–541 (2004).
9) 今枝直彦，小田原秀幸，政本廣志：属性証明書利用における属性証明書と公開鍵証明書のリンクに関する一考察，信学技報，ISEC2002-106 (2003).
10) 梅澤克之，坂崎尚生，佐藤一夫，松木 彰，曽我健二，片山 透，清本晋作，田中俊昭：モバイルサービス向け認証基盤の検討，電子情報通信学会論文誌 D，Vol.J90-D, No.2, pp.596–599 (2007).
11) 柿崎淑郎，辻 秀一：属性間のバインド方式の一検討，第 68 回情報処理学会全国大会，4E-6 (2006).
12) 柿崎淑郎，山本 宙，辻 秀一：属性認証を利用したプライバシ保護方式，情報処理学会論文誌，Vol.48, No.3, pp.1038–1046 (2007).
13) 千葉昌幸，漆嶌賢二，前田陽二：属性情報プロバイダ：安全な個人属性の活用基盤の提言，情報処理学会論文誌，Vol.47, No.3, pp.676–685 (2006).
14) 電子商取引推進協議会：属性認証の適用ガイドライン (2003).
15) 電子商取引推進協議会：証明書利用ガイドライン—属性情報の活用 (2004).

(平成 19 年 5 月 14 日受付)
(平成 19 年 11 月 6 日採録)

柿崎 淑郎（学生会員）

1980 年生．2003 年東海大学工学部電子工学科卒業．2005 年東海大学大学院工学研究科電子工学専攻博士課程前期修了．同年東海大学連合大学院理工学研究科総合理工学専攻博士課程進学．現在に至る．情報セキュリティ，プライバシ保護，属性情報の研究に従事．

辻 秀一（正会員）

1969 年大阪大学基礎工学部電気工学科卒業．1974 年大阪大学大学院基礎工学研究科博士課程修了，工学博士．1974〜2000 年三菱電機（株）に勤務．この間ヒューマンインタフェースや人工知能システム等の研究開発に従事．1997〜2000 年電子商取引実証推進協議会へ出向．2000 年 4 月より東海大学に勤務．現在，情報理工学部情報メディア学科教授．電子情報通信学会，人工知能学会，電気学会，IEEE 各会員．