トラストのためのデジタル署名検証

解説

2023 年 12 月 20 日

NPO 法人

日本ネットワークセキュリティ協会

電子署名ワーキンググループ

署名検証タスクフォース

目次

はじめに

さまざまなサービスがデジタル化し、ネットワークを介して提供され、デジタルデータが国境を越えて飛び交う中で、何をどうやって信用すれば良いかということは極めて重要かつ深刻な課題です。中でもデータについては、流通するデータ量が膨大になるにつれ、1つ1つを確認することは容易ではなく、信頼できる根拠が必要となります。それは、アナログ時代の「署名」や「押印」に対応するものとして、デジタルの技術である「電子署名」の役割と言えます。

署名は、それが付与されたデータを受け取った署名受領者が、確認して初めて効果があるものです。アナログの署名・押印であれば、目で見てそれなりの納得感（厳密には、サインの筆跡、印影の鑑定、封緘の確認等が必要ですが）を得ることもできますが、デジタルの場合、署名データを受領者が目視で確認することはできません。通常は、ソフトウェア（検証ツール）により「署名検証」を行い確認します。しかし、復号しないと利用できない暗号化データと違い、署名付きデータはそのままでも利用可能なため、署名検証しないで済ませることも多いのではないでしょうか。また、署名検証した場合でも、何を根拠としてその検証結果となったかまでは確認せず、検証ツールの結果を鵜呑みにしていないでしょうか。つまり、データが信頼できるか十分確認されないまま利用されている可能性があると言えます。

本書では、署名受領者に署名検証の確認観点を示すとともに、主にアプリケーション開発者（署名検証機能利用者）に署名検証の仕組みと注意点を理解してもらうことを目的とします。本書は、JNSA が発行した『デジタル署名検証ガイドライン』[1]に基づいて、主にトラストの視点から解説します。なお、本文中の図で出典を記していないものは、新規作成あるいは「デジタル署名検証ガイドライン」からの引用（一部修正）です。

[1] https://www.jnsa.org/result/e-signature/data/e-signature-guideline_v1.0_20210331.pdf

1　トラストと署名検証

1.1　トラストについて

ネット利用やデジタルの活用において、トラストが注目されるようになってきました。DFFT（Data Free Flow with Trust：信頼性のある自由なデータ流通）という概念も提唱されているように、データの流通においてトラストの重要性が高まっています。

“トラスト”は、日本語では“信用”とか“信頼”と訳されますが、辞書的には、“信用”は過去の実績の積み上げによる評価、“信頼”は互いに頼りになるという主観に基づく期待を意味するとされています。一方で、ここで扱う“トラスト”は、直接見えない相手とサービスを信頼し、そこから得られるデータを信用すべきかの判断を迫られるインターネット社会・デジタル社会において、安心、安全のための重要な要素です。つまり、何らかの仕組みや根拠に裏付けされて一定の信じられる状態が担保されること、その仕組みが提供されることと言えるでしょう。なお、類似の用語である“信頼性”（Reliability）は、『アイテムが，与えられた条件の下で，与えられた期間，故障せずに，要求どおりに遂行できる能力』（JIS Z 8115:2019『ディペンダビリティ（信頼性）用語』）として、機械やシステムの故障の発生しにくさを表す用語として用いられることに注意が必要です。

インターネット社会・デジタル社会において“トラスト”が必要となるシーンとしては、相手の認証、サービスの認証、データの認証などがありますが、本書では、データの“トラスト”を対象として、データそのもの及び、その作成者を確認することについて述べます。データのトラストを判断するには、誰が作ったものか、情報の中身は正しいかが重要です。しかし、中身の正しさは一般的な方法では判定できないため、ここではオリジナルが変更（改ざん）されていないことを要件とします。これらは、アナログの時代から署名・押印の役割であり、作成者と非改ざん性を検証することがデータのトラストの第一歩と言えます。

1.2　署名と署名検証の目的

署名あるいは押印は、印章の発生とともに洋の東西を問わずその持ち主の権威に紐づけられたり、意思と説明責任の表明、否認の防止といった用途に用いられてきました。また、印章は封緘に利用することで、内容の改ざん防止（改ざん検知）目的にも用いられてきました。これは、シール（Seal）として、自然人だけでなく組織等の文書等にも適用されています。

これらの、アナログにおける「署名」（手段としての手書きサイン、押印、封緘を含む）の目的を要約すると、文書等を作成、発信あるいは承認した人が署名を行った場合、

・作成者、発信者あるいは承認者（「署名者」と呼ぶことにします）を担保すること

・作成、発信あるいは承認されてから内容が変更されていないこと（非改ざん）を担保すること

と言えるでしょう。

また、それに対応する署名検証の目的は、

・署名者を確認すること

・署名後の非改ざんを確認すること

となります。アナログの署名の場合、その検証は、筆跡や印影の鑑定、紙やインクの状態から改ざん有無を判定するものでした。その判定根拠を簡単に下表にまとめます。

表 1.2-1　アナログの署名等における検証

	手書きサイン	押印	封緘
署名者の確認	本人のみが書ける前提、筆跡鑑定	本人のみが印章を持つ前提、印鑑の登録制度	本人/当該組織のみが印章を持つ前提、登録制度
非改ざん確認	紙やインクの状態で判断	紙やインクの状態で判断	封筒や封蝋の状態で判断

アナログにおける署名・押印の検証方法は必ずしも万全ではなく偽造も可能ですが、影響範囲がその書類に限られることなどから慣例的に用いられてきたと言えます。しかしデジタルにおいては、コピーや変更、流通が容易なことから、影響が広範囲に及ぶ可能性があり、より厳密な確認手段が必要とされます。また、ネットの向う側にいて見知らぬ人の身元の確認や、長期の時間経過後も確認できることが必要な場合もあるでしょう。

改ざんを防ぐためには物理的に強固なハードウェアで守る方法もありますが、データの流通の容易性、リアルタイム性などデジタル化のメリットを生かすためにもソフトウェアによる方法で対象のデータを保証する方法が有用です。デジタルデータに対する署名、いわゆる電子署名と署名検証の要件は、認証技術や暗号技術などを用いて、署名者を特定でき、1bit でも変更があると検出できることです。

2　電子署名と署名検証

2.1　各種の電子署名方式

デジタルデータに対する電子署名の方法にはさまざまなものがあります。代表的な方式として、以下の4つの類型について述べます。

(1) 認証記録型署名

アナログの署名に近く、なじみやすいものとして、例えばタッチペンを用いて手書きサインを入力したり、予め登録した印影や手書きサインの画像を対象のデータに貼り付けるものなどがあります。しかし、この行為だけで署名者確認と非改ざん性を担保することはできません。その行為をした人を認証技術等によって特定・確認し、その結果と署名対象データを紐づけて確実に記録・保管する必要があります。このような措置により電子署名の要件を満たすものを**「認証記録型署名」**と呼ぶこととします。

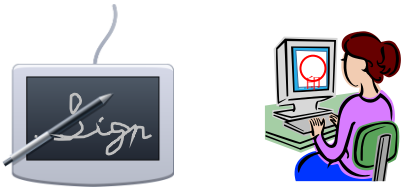

図 2.1-1　認証記録型署名のイメージ

(2) デジタル署名 (ローカル署名)

デジタルデータに対して、公開鍵暗号技術と公開鍵暗号基盤 (PKI：Public Key Infrastructure) を用いて実現される署名技術があり、これを**「デジタル署名」**と呼びます。デジタル署名は、署名者のみが持つ『署名鍵』というデータにより対象のデータに署名処理を施した署名データ (署名値) と、署名者がその署名鍵を保有すること (正確には、その署名鍵に対応する検証鍵に紐付けられた人であること) を証明する『電子証明書』により、非改ざんと署名者の確認を可能とするものです。

通常、署名鍵を本人が特殊な IC カード等で安全に保持し、本人の手元で署名を行うため、特に**「ローカル署名」**と呼ぶこともあります。

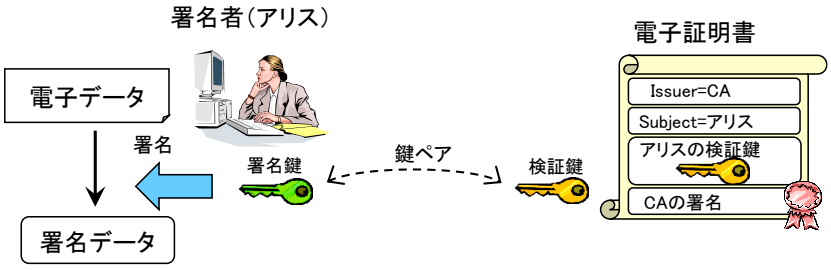

図 2.1-2　デジタル署名 (ローカル署名)

(3) デジタル署名 (リモート署名)

デジタル署名は、署名者が本人のみの鍵情報（署名鍵）を安全に保管（鍵管理）する必要がありますが、デジタルサービス普及に対する利便性、適切な鍵管理を実現するため、署名鍵を安全に預かり、署名を代行するサービスも出てきています。この形態を**「リモート署名」**と呼びます。（詳細は JT2A（日本トラストテクノロジー協議会）のリモート署名ガイドライン[2]参照）

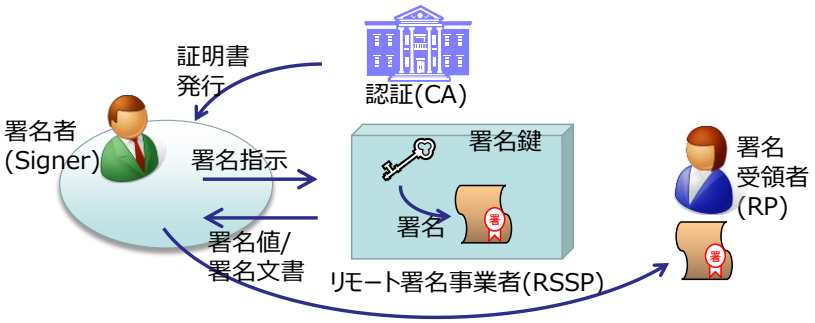

図 2.1-3 デジタル署名（リモート署名）（出典：リモート署名ガイドライン）

（4）事業者型署名

最近では、電子契約など、デジタル前提のサービスが増え、利用者間の契約等を仲介するサービス提供者（事業者）が、双方の利用者と内容を保証する形態として、**「事業者型署名」（立会人型署名）**という方式もあります。データの非改ざんの担保のため、事業者の署名鍵でデジタル署名されます。

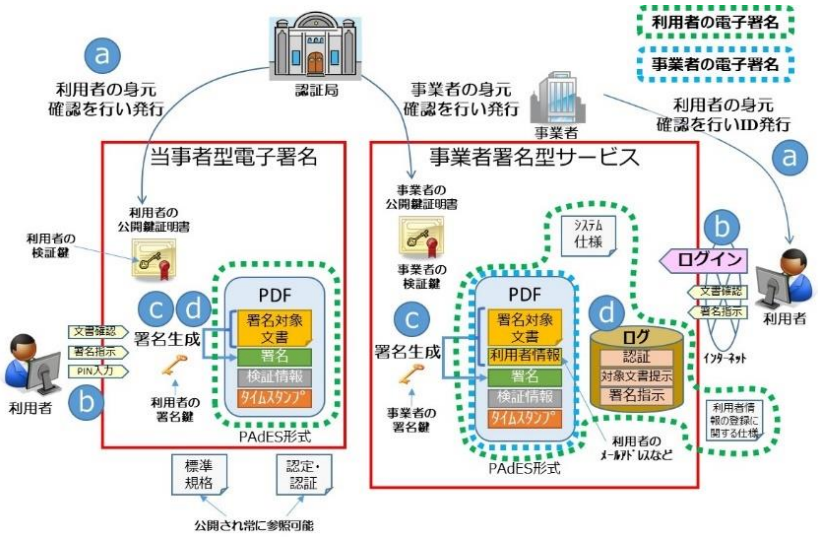

図 2.1-4 事業者型署名（出典：主務三省 Q&A（電子署名法第 3 条関係）に関する解説）[3]

[2] https://www.jnsa.org/result/jt2a/data/RemoteSignatureGguide_All-r1.pdf
[3] https://jdtf.or.jp/report/tsf/file/tsf/202102_主務三省 Q%26A_電子署名法第 3 条関係に関する解説.pdf、および https://www.c-a-c.jp/download/cmsassets/denshishomei_qa_kaisetsu.pdf

2.2　各種の電子署名における署名検証

いずれの方式においても、署名の用途が同じである限り、署名検証の目的は署名者と内容の非改ざんを確認することになります。

署名者の確認は、まさにその署名が表す人が誰であるか（身元）を確認すること、そして確かにその人が署名したこと（当人性）を確認することです。非改ざんの確認は署名対象データが、署名されてから確認時点まで改ざんされた形跡があれば分かること（改ざん検知）です。さらに、その後、任意の時点でも確認できるように、長期的に保証される仕組みがあることが望ましいと言えます。

認証記録型と事業者型においては、署名者の保証はサービス提供事業者の役割となり、非改ざん性の保証もその事業者が担うことになります（事業者型における、非改ざん性については事業者自身のデジタル署名で担保されます）。従って、事業者を信頼している利用者間で成立する署名方式と言えるでしょう。これは、信用・信頼を事業者に依存することとなり、事業者の事業撤退や廃業時の継続性については利用者のリスクととらえる必要があります。各種の署名方式における、署名者と非改ざん確認の主な根拠を下表にまとめます。

表 2.2-1　各種の電子署名における検証

		認証記録型署名	ローカル署名	リモート署名	事業者型署名
署名者の確認	身元確認	事業者による保証	認証局が発行した電子証明書	認証局が発行した電子証明書	事業者による保証
	当人確認	事業者による利用者認証	（本人のみが署名鍵を持つ前提）	事業者による利用者認証	事業者による利用者認証
非改ざん確認	改ざん検知	事業者による記録を確認	本人のデジタル署名を検証	本人のデジタル署名を検証	事業者のデジタル署名を検証
	長期の保証	事業者の継続	長期署名[*1]を検証	長期署名[*1]を検証	事業者の継続

*1：長期間の保証を可能とするデジタル署名方式（4 章参照）

これらの署名方式は、保証したいデータの重要性や用途に応じて選択して利用することになりますが、方式が標準化され公開されているものは、保証の根拠がより客観性を持つと言えるでしょう。以降、本書では、標準化された技術に基づく PKI ベースのデジタル署名について述べます。

3　デジタル署名における署名検証

3.1　デジタル署名における基本の検証処理

デジタル署名は標準のフォーマットが国際的に規定されています。バイナリデータや、XML データ、PDF データなどいくつかの形式がありますので、詳細は、RFC 5652/5126、ISO 14533 シリーズ、電子文書長期保存ハンドブック（次世代電子商取引推進協議会,2007.3）[4]等を参照してください。下図に一例として、バイナリデータに対する署名データの基本の形を示します。

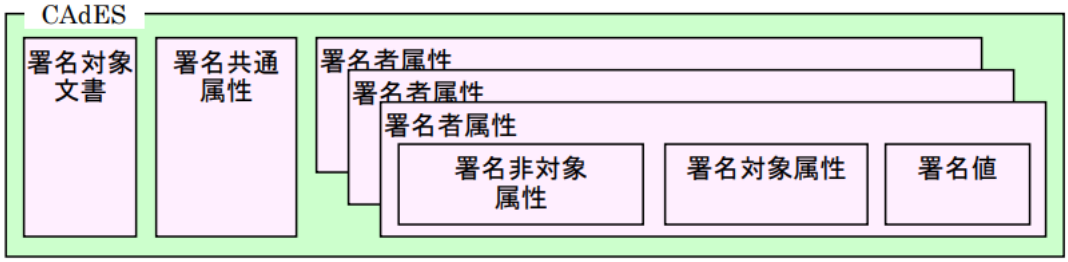

図 3.1-1 署名データの基本構造（出典：電子文書長期保存ハンドブック）

これらのデータに対する検証処理としては、データ構造を確認した後、構成要素に対して、

・署名値検証

・証明書検証（認証パス検証、有効性確認を含む）

を行います。なお、タイムスタンプを伴う署名データについては、4 章で述べます。

実際には、これらの処理については様々なライブラリが提供されているため、詳細な処理内容を熟知していなくても、データフォーマットや用途に応じて適切なものを選定して利用することができます。本章では、非改ざんを確認する署名値検証と署名者を確認する証明書検証の基本について簡単に説明します。

3.1.1　署名値検証

データが改ざんされていないことの確認は、署名値を検証することで行えます。署名値検証は、アナログの署名においてサインや押印された紙に改ざんの跡が無いか、封緘された書類が開封された様子が無いかを確認することに相当します。

公開鍵暗号技術では、特定のアルゴリズムにより署名処理、検証処理を行います。そのアルゴリズムには、ある秘密鍵で署名したものは、それと対になる公開鍵でしか検証できないという性質があります。デジタル署名においては、秘密鍵を署名者のみが持つ『署名鍵』、それと一対の公開鍵を『検証鍵』と呼びます。

[4] https://www.jipdec.or.jp/archives/publications/J0004262.pdf

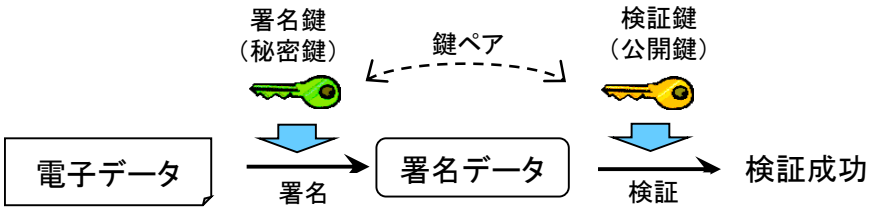

図 3.1.1-1 公開鍵暗号技術による署名・検証

通常、署名対象データのハッシュを作成して、それに対して署名鍵で署名処理を行って署名値を生成します。受領者における署名検証は、署名値を検証鍵で検証処理し、署名対象データのハッシュと比較して判定します。(詳細は各署名アルゴリズムの説明を参照してください。)

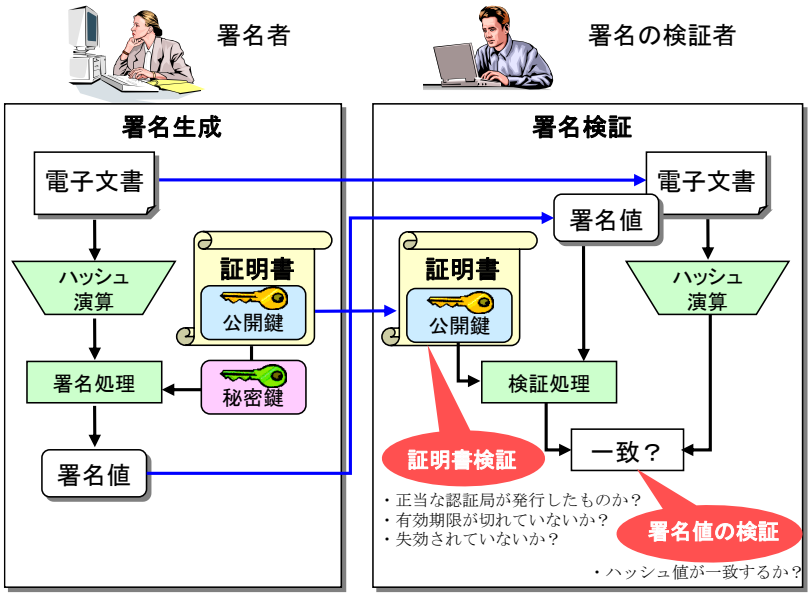

図 3.1.1-2 公開鍵暗号技術による署名と署名検証

署名鍵は署名者が安全に保管し、署名者固有の署名を生成します。一方、検証鍵は公開して署名を検証したい人が使うことができるようにすることで、誰でもその署名を検証し、署名してからデータに変更(改ざん)がなされていないことが分かります。

なお、後述の証明書検証にて、署名鍵の有効期間が終わっていたり、失効していないことが確認されることが前提です。

3.1.2 証明書検証

アナログの場合は、筆跡や印影を登録簿などと照らし合わせて署名者・押印者を確認することになりますが、デジタル署名の場合、署名者は電子証明書によって確認します。

署名値検証により、非改ざんが確認されたとしても、その署名鍵が誰のものかが分からなけれ

ば署名の検証にはなりません。署名鍵を署名者が安全に保管し、検証鍵を署名者と紐づけて公開しておけば、それを用いて誰でも署名者を確認することができます。その検証鍵と署名鍵の持ち主を紐付ける証明書が電子証明書（公開鍵暗号技術では一般に公開鍵証明書と言います。以後、単に証明書と略します）で、その証明書を’信頼できる第三者’としての『認証局』（CA；Certification Authority）が発行するスキームがあります。日本では実印を印鑑登録すると、その印影を証明する印鑑証明書が自治体により発行されるのと類似のスキームと言えます。証明書検証は、その証明書を確認し、署名者が誰であるかを知ることに相当します。

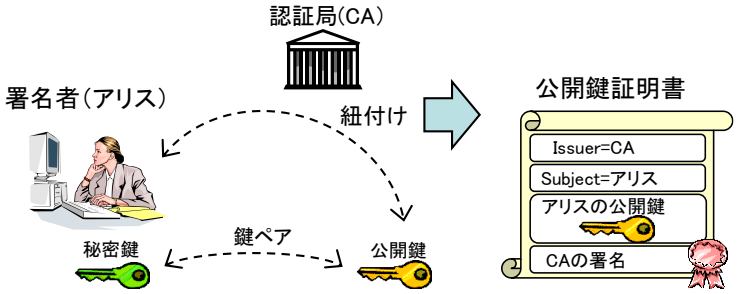

図 3.1.2-1 認証局と公開鍵証明書

（1） 認証パス検証

証明書のフォーマットは国際的な標準として規定されており、一定の技術があれば誰でも作成することができます。証明書にはそれを発行した認証局の署名が付与されますが、証明書を発行する無料の認証局サービスや、署名を生成する無料のツールもあります。つまり、誰かになりすまして技術的には正しいデジタル署名と証明書を生成することができてしまいます。

そこで、証明書を発行した認証局がどういう認証局かを保証するため、より上位の認証局が下位認証局の証明書を発行します。さらにその上位の認証局が保証するという連鎖が発生し、最上位の認証局はルート認証局と呼ばれます。この連鎖を認証パス、連鎖した証明書を証明書チェーンと言います。ルート認証局あるいは途中の認証局（中間認証局）が信用できるものであれば、その証明書も信用できると考えられます。

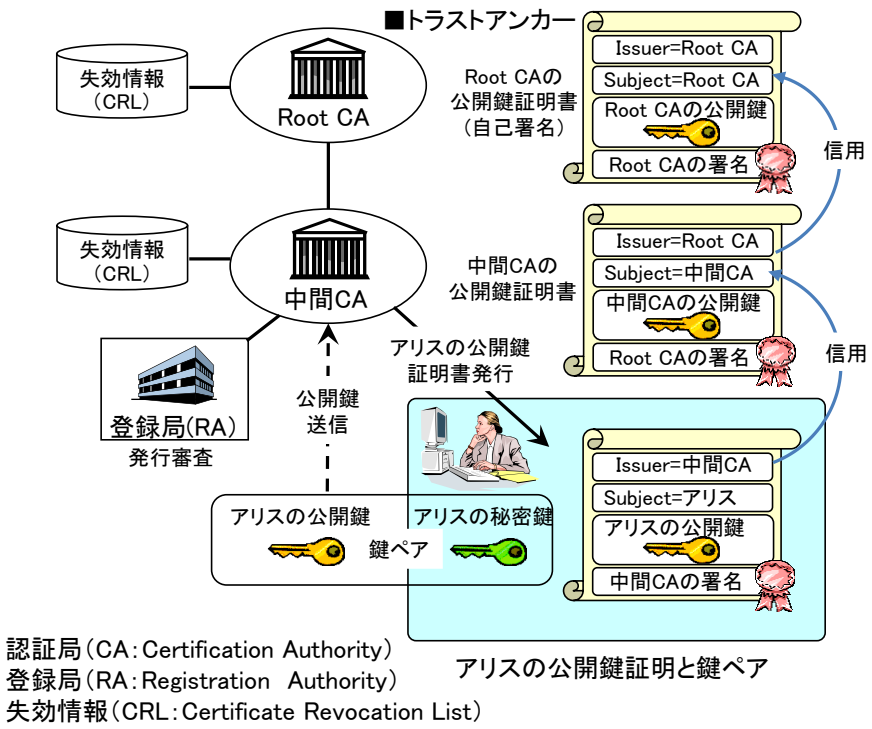

図 3.1.2-2 認証局の階層構造と公開鍵証明書

証明書の検証は、「認証パスの構築 (Certification Path Construction)」と「認証パスの検証 (Certification Path Validation)」の手順で行います。

認証パスの構築は、検証したい証明書から出発して信用関係を順にたどり、自分が信用している CA まで結ぶものです。信頼の基点となる CA を「信頼点(トラストアンカー: Trust Anchor)」と呼びます。

認証パス検証は、上位認証局の署名を検証しながら、トラストアンカーまで証明書の連鎖を確認することになります。

(2) 有効性検証

公開鍵暗号技術の重要な注意点は、署名鍵には有効とされる期間に限りがあることです(これは、公開鍵暗号技術が計算量的な安全性に基づいていることに起因します)。このため、署名鍵と対になる検証鍵を保証する公開鍵証明書には有効期間が設定されています。従って、証明書を確認する際には、署名されてから、検証するまでの間に有効期間が終わっていないかを確認することが必要です。

また、署名鍵を署名者が安全に保管していることが大前提であると述べた通り、署名鍵を紛失したり盗難にあった場合には、クレジットカード等と同様に、速やかに失効届けを出す必要があります。公開鍵暗号基盤PKIでは失効したものをCRL(Certificate Revocation List:証明書失効リスト)等の方法により情報提供されます。証明書検証の際には、この情報にアクセスして失効して

いないことも確認する必要があります。

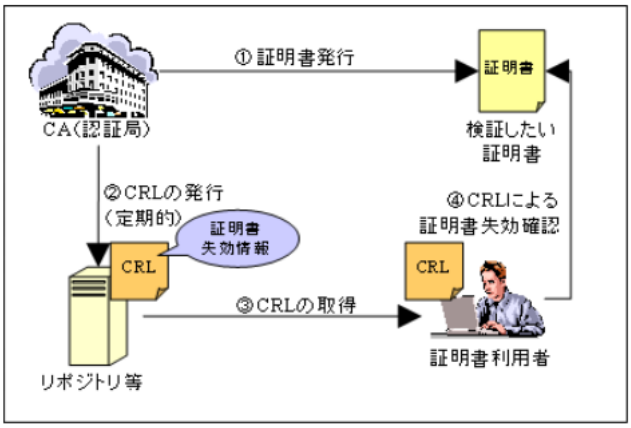

図 3.1.2-3　証明書の失効(出典:「PKI 関連技術情報」[5])

(3) 署名者の確認

以上の処理を経て、署名値検証に成功した検証鍵の証明書に記載された人が署名者と分かります。まとめると、以下となります。

① 署名文書の利用用途に応じた適切な証明書を用いていたこと
② 署名当時に証明書の有効期間内であったこと
③ 失効していない証明書を用いて署名していたこと
④ 署名文書の利用期間を通じて、上記①～③が確認可能であること。

下図では①から③を図示していますが、より厳密には①及び②に関して、署名時刻がいつであったのか客観的に示すためにタイムスタンプが利用されること、また署名時点での証明書の有効性を確認するために失効情報が保管されていることが必要です。

[5] https://warp.ndl.go.jp/collections/info:ndljp/pid/12308150/www.ipa.go.jp/security/pki/

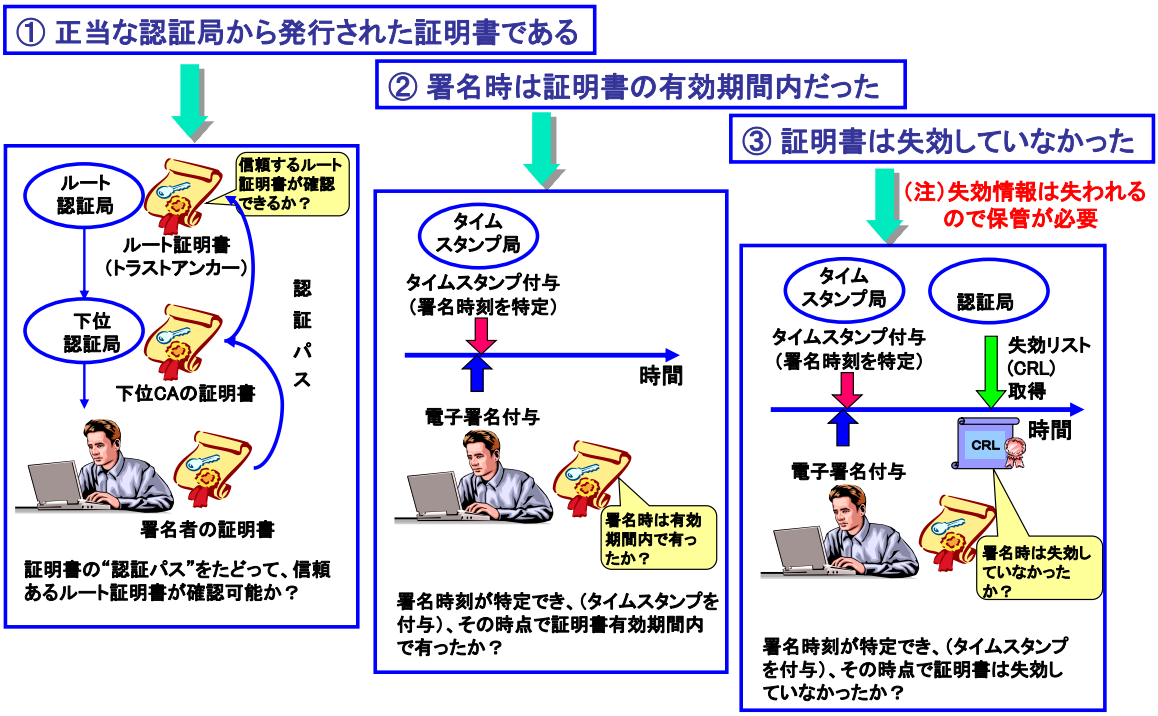

図 3.1.2-4 証明書検証の要素

3.2 トラストに係る検証要素

前節で基本の署名検証について述べましたが、これらはデジタル署名の技術に基づき、機械的に処理できるものと言えます。ところが、実際のビジネス利用を想定したとき、トラストの確認はこれだけで十分とは言えません。

署名の目的である署名者の確認と非改ざんについて、前節で確認できることは下記でした。

［署名者の確認］

① 署名者が検証に成功した証明書に記載されていること

② 証明書の発行元がルート認証局（又は信頼できる認証局）までたどれること

③ これらの証明書が有効期間内であり、失効されていないこと

［非改ざんの確認］

① 署名対象データの署名値が証明書に含まれる検証鍵で検証できること

② 署名鍵が有効である（証明書の期限切れ、失効がない）こと

これらの確認において前提としていたこと、当然と思われがちなことがあります。署名の用途にもよりますが、トラストのためには以下のことも確認する必要があります。

署名者の確認においては、

i) 証明書のトラスト：認証パスとルート認証局は本当に信頼できるか（トラストアンカー）

ii) 署名者のトラスト：署名者は実在の（身元が確認された）本人か（本人性）

iii) 有効性のトラスト：有効性、失効情報は信頼できるか

非改ざんの確認においては、上記の有効性に加えて、

iv) 暗号アルゴリズムのトラスト：本当に改ざんが不可能か（脆弱性、危殆化の有無）

v) 署名鍵のトラスト：署名鍵は安全に保管されていたか（鍵管理）

また、その他共通的な要素として、

vi) 時刻情報のトラスト：署名時刻などの時刻情報は信頼できる正しい時刻か

vii) 検証ツールのトラスト：署名の適切な処理、特に各種制約条件に適合しているか

もあります。以下では、これらについて述べます。

3.2.1 証明書のトラスト

前節で、証明書の連鎖である認証パスを検証することは述べました。しかし、最上位のルート認証局はその認証局自身が署名した自己署名証明書で保証されています。また、どのように連鎖が形成されているか、ルート認証局がどのようなものかを一般の署名受領者が知っているとは限りません。このような場合に、何に基づいて信用を判断すべきでしょうか。

（1）認証局のトラストモデル

一般的に PKI のモデルは、階層型の認証局の構成で説明されます。署名者、検証者の双方

が共通の認証局を信頼できる場合、認証パスが繋がったということになります。また、それぞれの認証パスが共通のルート認証局にたどり着けない場合でも、認証局同士が相互に保証し合う相互認証(bilateral cross certification)型の構成を持って連携することがあります。また、複数の認証局が相互認証するメッシュ型や1つの認証局をハブとして仲介して形成するブリッジ型の認証構成を取る場合もあります。

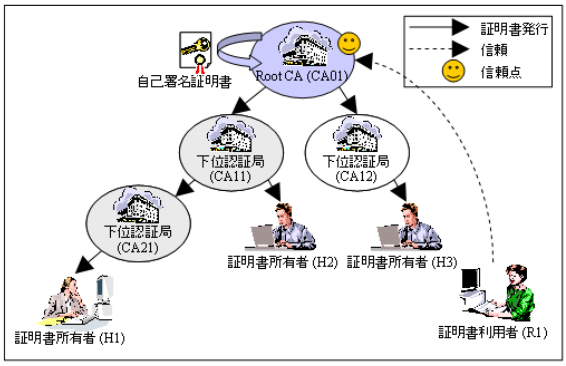

図 5-6 階層型モデル

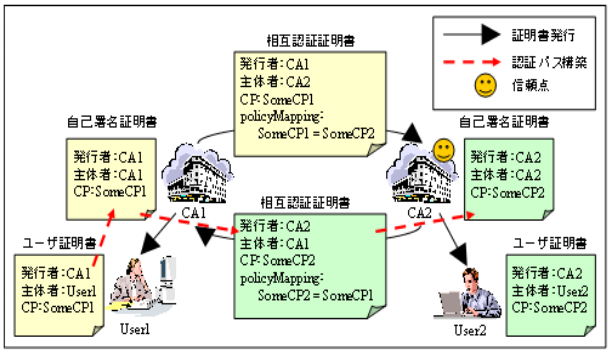

図 5-9 相互認証証明書を用いた認証パスの構築

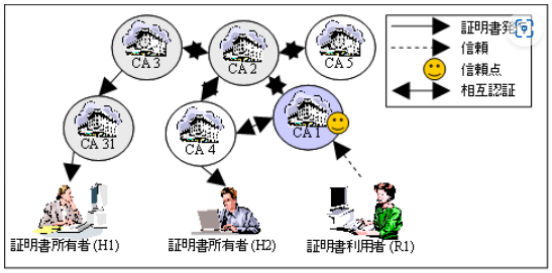

図 5-10 メッシュモデル

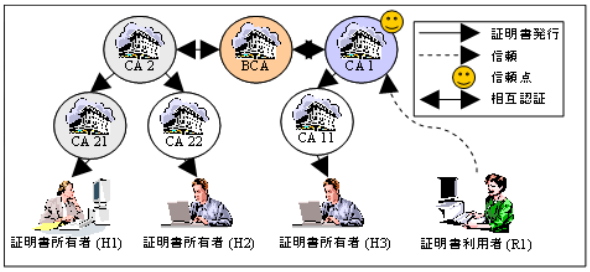

図 5-12 ブリッジCAモデル

図 3.2.1-1 トラストモデル(出典:「PKI 関連技術情報」)

ここで、ルート認証局の確認方法として、自己署名証明書の検証だけでなく、その証明書のハッシュ情報(FingerPrint)が信頼できる情報源(官報など)に公開され、誰でも確認できるようにしている場合もあります。

また、ルート認証局が信頼できるかを一般の利用者が都度手作業で確認することは非常に困難です。そこでインターネット利用の際、大抵の利用者が使う Web ブラウザーに「信頼できるルート証明書」をセットしておき、ブラウザーが自動的にそれをチェックする Web 型トラストモデルもあります。

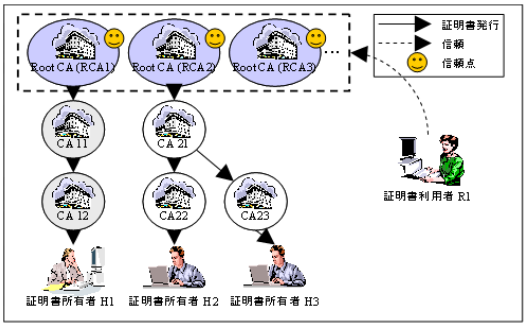

図 5-7 Webモデル

図 3.2.1-2 Web モデル（出典：「PKI 関連技術情報」）

これは、ブラウザーに一定基準を満たす証明書がルートとして組み込まれており、それと認証パスが繋がることで、信頼できるとする方法です。

なお、認証局は単に連携すればよいというものではありません。認証局同士の運用や証明書の発行方針が整合している必要があります。また、認証パスで許容される連鎖の数の制限もあります。証明書や運用のポリシーについては認証局が発行・公開する CP/CPS（Certificate policy/Certification Practice Statement：証明書ポリシーおよび運用実施規程）などで確認できます。

(2)トラストアンカー

信頼の基点となるものがトラストアンカーです。特に重要な取引においては、より厳格にトラストアンカーを確認すべきでしょう。当然、署名する際にも同様の考慮により、何をトラストアンカーとするかを十分検討して選定することが重要です。コストの観点だけで認証局を決定することは適切ではありません。

Web モデルは便利な方法ですが、ブラウザーに組み込まれたものを常に信用してよいでしょうか。著名なブラウザーには非常に多数の「信頼できるルート証明書」がセットされており、追加削除も利用者が行うことが可能ですが、すべてのルート証明書を確認して適切に管理することは容易ではありません。ある時点で「信頼できるルート証明書」として登録されていても、インシデント等により信頼できなくなるケースも発生しています。あくまで、一般のネット利用のレベルで活用するものとも言えます。

過去には、認証局がサイバー攻撃により不正操作され、信頼できない証明書が発行されたこともありました。証明書が意図せず発行されたものでないことを確認するための CT(Certificate Transparency)という仕組みも一部で運用されています。

なお、欧州ではトラストサービスを登録・管理するトラステッドリスト（TrustedList）というスキームがあり、トラストサービスの1つである認証局も一元的に管理されており、現時点だけでなく、過去にさかのぼり、有効性を確認できます。

3.2.2 署名者のトラスト

証明書に記載された人は、本当に実在し、また署名受領者が想定し、期待する人と同一人物でしょうか。署名者が確かに証明書に記載されている当人（署名鍵の所有者）かどうかは認証局がその人を確認して証明書を発行するプロセスに依存します。

ローカル署名の場合は、当人が署名鍵を安全に保管している限り、署名検証をすることで確認できます。リモート署名の場合は、署名者がリモート署名事業者に署名依頼する際の認証によって担保されることになります。

（1）認証局の役割

認証局の機能構成は一般に、利用者を確認して登録する登録局（RA：Registration Authority）と、証明書を発行する発行局（IA：Issuing Authority）から構成されます。RA にて利用者の身元を確認する方法は、メールの到達性に基づき自動的に行うものから、人手により公的書類等を確認するもの、何らかの組織のデータベースに基づくものなどさまざまです。認証局のビジネス上の差異化要素でもあり、また、法律等により、基準が設定されているものもあります。その結果として、身元確認レベル（署名者の身元がどの程度信頼できるか）も異なります。

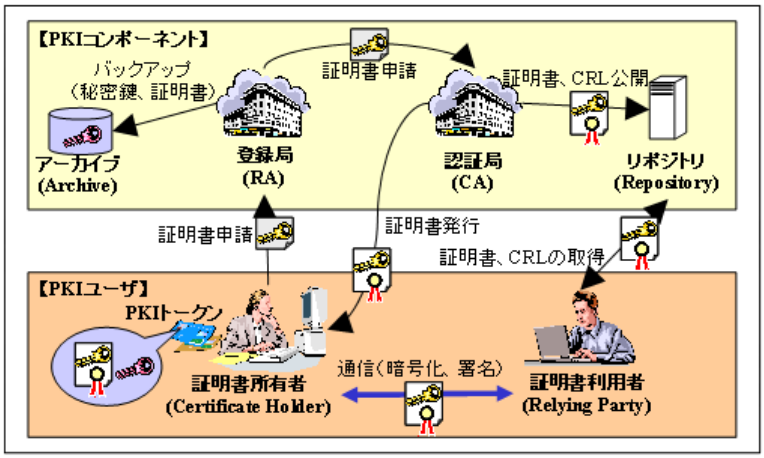

図 3-4 PKI 構成要素

図 3.2.2-1　認証局の構成（出典：「PKI 関連技術情報」）

（2）身元確認レベルの確認

一般には、認証局の発行する CP/CPS により、どのような確認に基づいて証明書を発行しているかを確認できます。署名の用途に応じて、相応しい身元確認が行われた証明書かを確認することが望ましいと言えます。特に無料で発行された証明書などでは、相手の身元が保証されないことが多く、テスト用途に用いることはあっても、実ビジネスでの利用は慎重であるべきでしょう。

3.2.3 失効情報のトラスト

前述の通り、証明書は有効期間内でも署名鍵の紛失、盗難、漏洩等によって失効することがあります。失効後の署名は必ずしも信用できないため、失効情報を確認する必要があります。これは、利用者の証明書に限らず、証明書を発行した認証局の証明書についても同様です。ここで、失効情報を確認する場合にも注意点があります。

(1) 失効情報の種類

失効した証明書は、リスト(CRL)にして認証局が提示します。失効確認手段としては、CRL、OCSP(Online Certificate Status Protocol)などがあり、それらの情報を提供するサービスを適切に利用する必要があります。CRL は一定期間の間に失効申請のあったものを認証局がリストにして公開し、利用者がリストを取り寄せて確認したい証明書が含まれていないかチェックします。公開の周期は認証局により異なりますが、業界や用途により定められている場合もあります。OCSP はある証明書が失効していないかを問い合わせて確認できるものです。

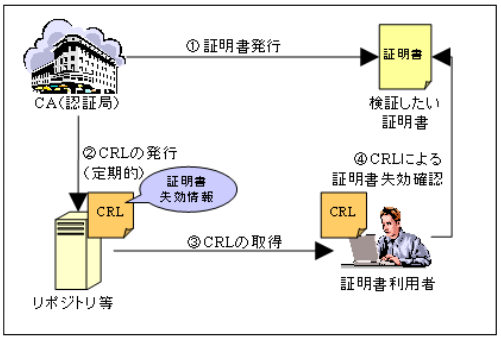

図 4-1 CRL モデル概要

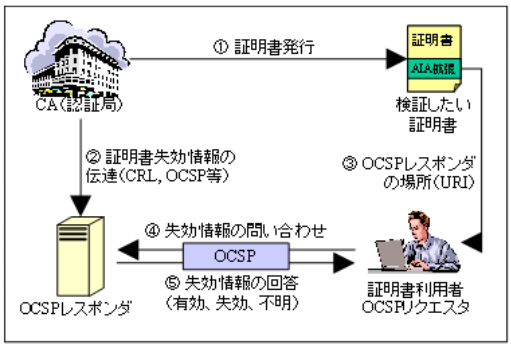

図 4-9 OCSP 概要

図 3.2.3-1 失効情報の確認方法(出典:「PKI 関連技術情報」)

(2) 失効確認の注意点

いずれの方式も、失効(特に鍵の漏洩などの場合)の発生からタイムラグがあることに注意が必要です。例えば、①漏洩してから発覚するまでの時間、②発覚してから失効申請するまでの時間、③失効申請されてから CRL に掲載され発行されるまでの時間(最大が CRL 発行周期)、④CRL が発行されてから取得して確認するまでの時間があります。OCSP においても、CRL に基づいて応答する場合には同様のタイムラグがあります。

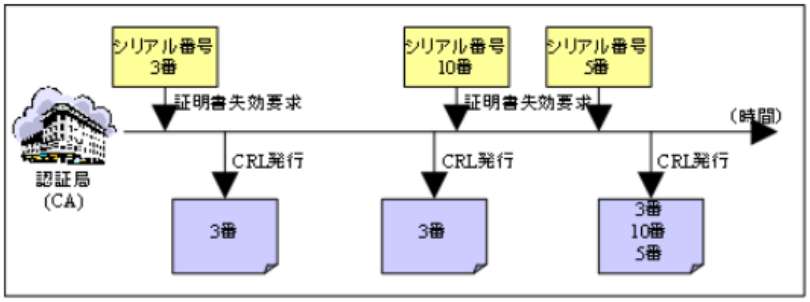

図 3.2.3-2 CRL の定期的発行（出典：「PKI 関連技術情報」）

また、タイムラグを想定して検証する（検証が許容される）タイミングを決める考え方があります。それは、猶予期間（Grace Period）を置いてから検証するもので、後述する長期署名などのように署名後、タイムスタンプや再度の署名を行うタイミングとしては、元の署名から適切な時間間隔を置くことが必要です。

すなわち、署名やタイムスタンプ生成後、猶予期間を経ていること、次のタイムスタンプ付与又は検証の時点で、失効情報の発行周期以内で最も新しい（鮮度が高い）ものであることが求められます。なお、実際には、ルート認証局や中間認証局の失効情報、OCSP のタイミングなど、状況に応じて考慮が必要となります。

3.2.4 暗号アルゴリズムのトラスト

デジタル署名の原理では、署名鍵で署名したものは、それと対になる検証鍵でしか検証できません。つまり、改ざんされているか、検証鍵が違っていれば検証エラーとなり、エラーがなければ改ざんされていないと考えられます。ただし、この場合、非改ざん性を確認するためには、証明書が失効していないこととともに、アルゴリズムが危殆化していないことが条件です。

（1） アルゴリズムの危殆化

前述の通り、現代暗号技術は計算量安全性に基づくものが主流で、技術の進歩、時間の経過とともに、安全性が低下し、脆弱性が増加します。アルゴリズム自体の解法の進歩だけでなく、計算能力の大幅な向上により安全でなくなった場合、危殆化したと言われます。デジタル署名に関しては、署名アルゴリズムだけでなく、ハッシュアルゴリズムについても同様です。

技術の進化とともに計算能力は年々向上しています。図 3.2.4-1 に暗号技術検討会 CRYPTREC による、暗号アルゴリズムの計算困難性の一つである素因数分解に要する計算量と、年ごとの計算能力の関係を示します。どの時点で安全でなくなるとするかは、いろいろな見方があると思いますが、暗号技術検討会の発表などを参考にしてください。

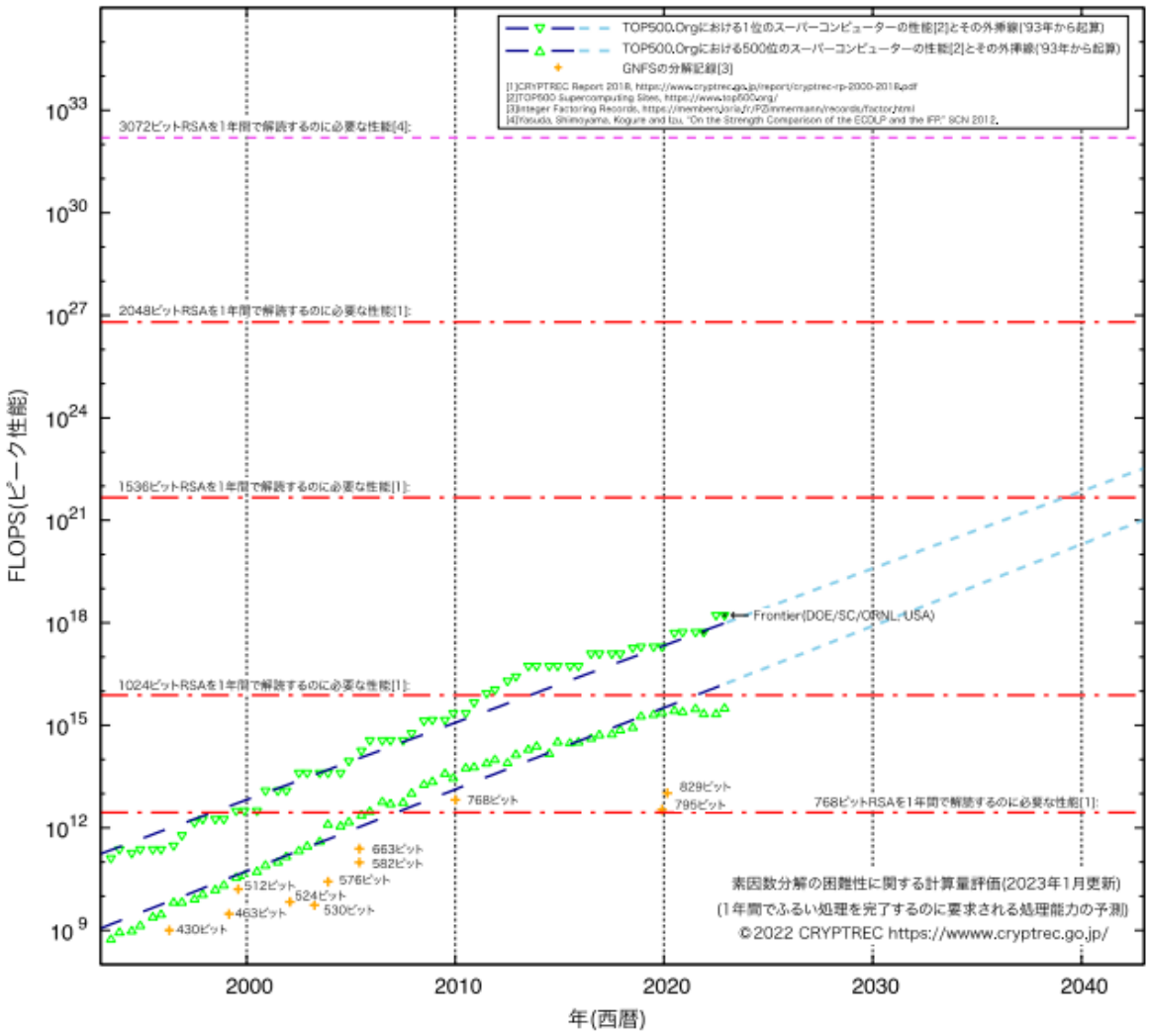

図 3.2.4-1　素因数分解の困難性に関する計算量評価(2023 年 1 月更新)
「暗号技術検討会 2022 年度報告書」[6]より

(2) 脆弱性・危殆化に関する情報

署名データには各種暗号アルゴリズムが用いられ、その種別を識別する情報は署名データに含まれています。ところが、各暗号アルゴリズムが利用された時点で脆弱でなかったことを示す根拠は署名データには含まれていません。従って、各暗号アルゴリズムが利用された時点で脆弱でなかったことを確認するためには外部の情報を参照する必要があります。例えば日本においては、CRYPTREC 暗号リスト(電子政府推奨暗号リスト)は確認すべきでしょう。

実際には、暗号アルゴリズムの利用箇所は多岐にわたるとともに、その安全性の基準等が明確でないため、何らかの制約を設けない限り確認は困難です。その課題はデジタル署名検証ガイドライン「付属書 C」に述べられています。

[6] https://www.cryptrec.go.jp/report/cryptrec-rp-1000-2022.pdf

また、同じアルゴリズムでも鍵の長さによって安全性は変わります。用途に応じて適切な鍵長を選択する必要があります。CRYPTREC の「暗号強度要件（アルゴリズム及び鍵長選択）に関する設定基準」[7]などが参考となります。

3.2.5 署名鍵のトラスト

デジタル署名は、署名鍵が安全に管理されていることが大前提となっています。しかし、これは簡単なようで意外と難しい条件とも言えます。

（1） 署名鍵の安全な管理

所有者が意図的に漏洩させることは除くとしても、盗難等に備えて安全に管理する必要があります。その1つの方法が、耐タンパー性（外部から不正にデータを読み取られたり、改ざんされることに対する耐性）のあるデバイスに保管することです。認証局などの事業者は HSM（ハードウェアセキュリティモジュール）を用いますが、一般の利用者は耐タンパー性のある IC カードを用いることが多いでしょう。PC 等に保管する場合、意図せずエクスポート（取り出し）されない対処が必要です。

運用としては、有効期間を確認して更新すること、紛失、盗難時などには確実に失効申請することが必要です。

検証者は、失効情報の確認以外に署名者が安全に署名鍵を管理していたかを検証する手段はありません。用途によっては、耐タンパーデバイスの適用を必須とする取り決めなどが必要でしょう。

（2） リモート署名

これらの鍵管理の運用の煩雑さを解決する方法として、署名鍵を預かるサービスがあります。これがリモート署名と呼ばれる方式で、署名鍵を安全に管理しつつ、リモートで署名を代行するサービスと言えます。欧州などで、リモート署名の基準が整備されつつあります。（附録 A.5 参照）

3.2.6 時刻情報のトラスト

署名内容の日付に関する情報以外に、デジタル署名においては、署名時刻、証明書の有効期間など、種々の時刻情報があります。これらの時刻情報のトラストが重要になります。

（1）時刻情報源

デジタル署名を行うサーバーや PC、モバイル端末は、通常、時計を内蔵し、その時刻情報で処理します。インターネットで時刻合わせを行うこともありますが、逆に、時刻を設定変更して動作させることも可能です。署名者の署名環境の時刻が正しいかどうかを、検証者が確認することはで

[7] https://www.cryptrec.go.jp/list/cryptrec-ls-0003-2022r1.pdf

きない以上、時刻情報を信用できるかどうかは用途の重要性に応じて考える必要があります。

(2) デジタルタイムスタンプ(以降、タイムスタンプと呼ぶ)

時刻を、署名者、検証者以外の第三者が保証する仕組みとしてタイムスタンプという仕組みがあります。タイムスタンプには幾つかの方式がありますが、シンプルプロトコル(RFC 3161)による方式が主流です。

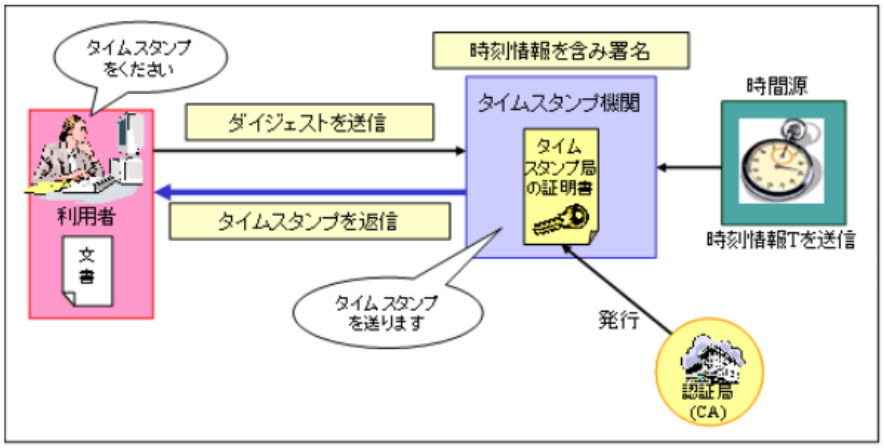

図 3.2.6-1 タイムスタンプの仕組み(出典:「PKI 関連技術情報」)

署名データ内の時刻情報(SigningTime)などに頼らず、署名データにタイムスタンプを付与することで時刻の客観性を保証することができます。時刻の厳密性を必要とする場合や、長期に署名データを保証したい場合に使われます、これを用いた署名のフォーマットについては 4 章で述べます。

3.2.7 検証ツールのトラスト

本書の冒頭で述べたように、デジタル署名を人手で検証することはほぼ不可能で、何らかのツールを用いることになります。そのツール(署名検証アプリケーション)が用途に応じて適切な検証処理を行えることが重要です。

(1) 各種の制約条件

署名の利用用途に応じて、使い方や扱いに制約を加えることがあります。適用領域や法制度等の要請により、標準規約で検証必須として規定されている要素の検証を不要としたり、逆に検証オプションとして規定されている要素の検証を必須としたりする場合もあります。

その場合、署名検証アプリケーションは、検証対象となる署名データ(署名対象のコンテンツを含む)だけでなく、外部からの情報を参照する必要があります。また、署名利用分野の必要に応じて、制約条件を与えることがあります。これらの情報を総称して検証制約と呼びます。

検証制約の与え方としては以下の方法があります。

- 署名ポリシー（標準規約準拠）
- 設定ファイル（独自形式）
- 実装ロジックへの埋め込み

（2） 適合宣言

トラストのためには、署名検証を行うツールやアプリケーションがそれらの制約を適切に処理できるとともに、どの制約にどう対応しているかを明らかにすること、そして利用者（署名検証者）がその情報に基づいて適切なツールやアプリケーションを選択して利用する必要があります。

例えば、デジタル署名検証ガイドラインでは適合宣言書を規定しています。基本的な要素から特定の分野における制約まで、ツールの提供者は処理条件を明確化し、それを利用者に分かり易く提示する必要があるでしょう。

4　長期的な署名データの検証

4.1　長期署名方式

デジタル署名には有効期限切れや危殆化のリスクがあります。ある程度以上の長期的な署名の効果を期待する場合には避けて通れない課題であり、その対策として、長期署名方式 AdES（Advanced Electronic Signature）が規定され、標準化されています。AdES とは、有効期限が切れる前にタイムスタンプを付加することで、有効性を延長していく方式です。

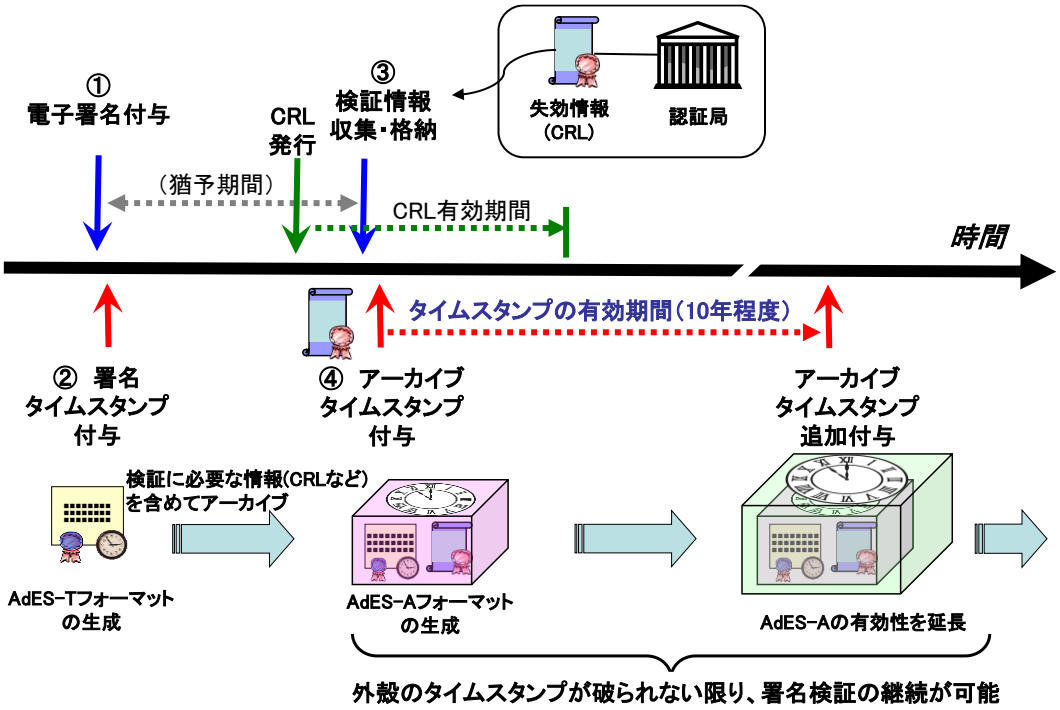

図 4.1-1　長期署名フォーマットによる署名延長

AdES 方式による署名データのライフサイクルを下図に示します。

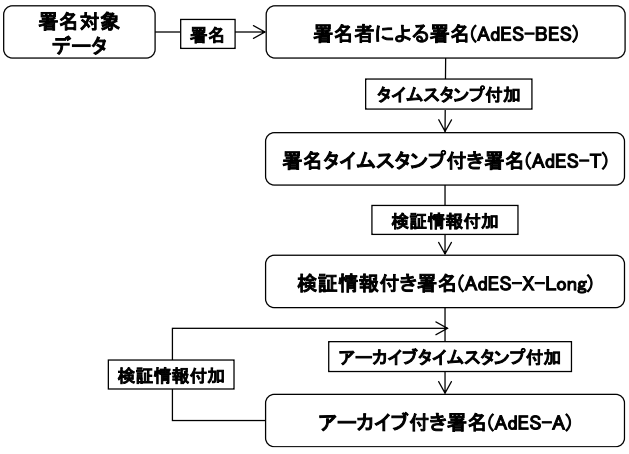

図 4.1-2　署名データのライフサイクル

4.2　長期署名の検証

AdES データの基本の検証処理は、前章と同様に、構造を解析して、証明書や署名値に加えて、タイムスタンプの検証を進めていくことになります。

4.2.1 検証プロセス

AdES データの場合、上記ライフサイクルの通り、AdES-T 以降、タイムスタンプとその発行元であるタイムスタンプ局の証明書が含まれます。

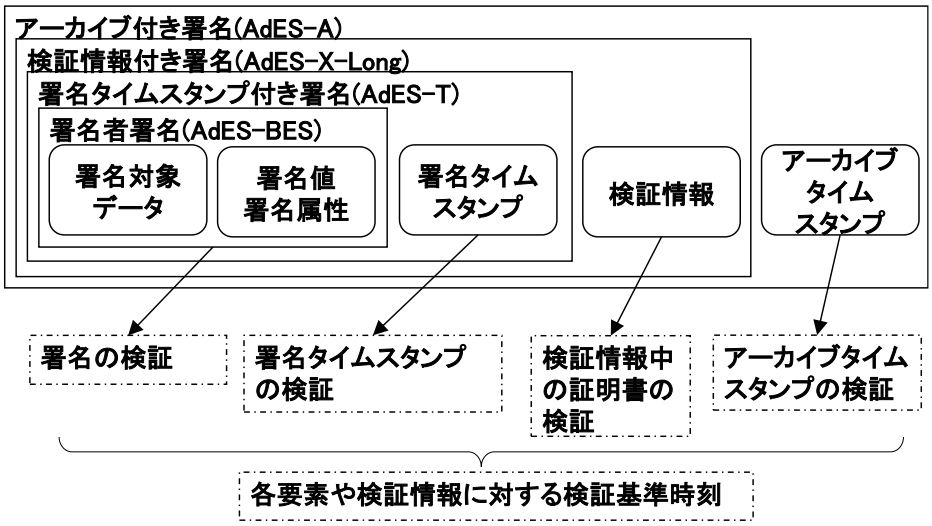

図 4.2.1-1 長期署名フォーマットの検証

タイムスタンプや、タイムスタンプ局の証明書についても、デジタル署名検証ガイドラインにある通り、検証を行うことになります。その検証のポイントとして、タイムスタンプの検証と、検証の基準時刻について述べます。

4.2.2　タイムスタンプの生成と検証

RFC 3161 に基づくタイムスタンプも署名や証明書と同様、標準技術のため、一定の技術があれば誰でも生成できます。従って、証明書に対する認証局と同様、タイムスタンプの信頼性を担保するためタイムスタンプ局（TSA：Time Stamping Authority）があり、そのタイムスタンプ局（以後、TSA）を認証する認証局があります。

タイムスタンプの生成プロセスは、対象データのハッシュ値を、第三者である TSA に送付し、TSA は、信頼のおける時刻を付与し検証に必要な情報を加えたタイムスタンプ署名対象データ（TSTInfo）を生成します。TSA は、TSA が管理している署名鍵でデジタル署名したタイムスタンプトークン（TST）を生成し、TST を含むデータをレスポンスとして利用者に返します。

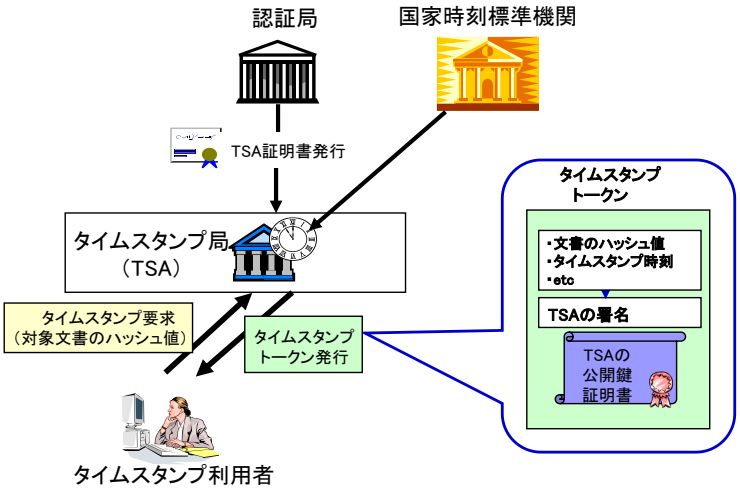

図 4.2.2-1 タイムスタンプトークンと TSA の役割

タイムスタンプの検証は、以下の流れとなります。

① TSA 公開鍵証明書及びその認証パスの検証
② TST の署名値を TSA 公開鍵証明書に含まれる TSA 公開鍵で検証
③ TST に含まれる TSTInfo から、タイムスタンプ署名時刻（genTime）とタイムスタンプ対象情報を抽出
④ タイムスタンプの対象データと TST に含まれる messageImprint との一致確認

トラストの観点で言うと、適用領域や法制度の要請等により、信頼すべきタイムスタンプサービスを選択する必要があります。検証時に信頼すべきタイムスタンプであるか否かを判断するために、タイムスタンプトークンに含まれる署名時刻源、タイムスタンプポリシー、発行者、信頼点、精度等の要素に関する制約を外部から与える場合もあります。（詳細は附録 A.7 で述べます。）

4.2.3 検証基準時刻（validation reference time）

長期署名方式の署名データの検証において注意しなければならないのが、検証時刻です。署名検証は、デジタル署名を付与し、一定期間経過した後に行われる行為であることに着目してみると、署名の有効性を確認する時刻の設定によっては、証明書の失効や暗号アルゴリズムの脆弱化などの要因により、検証結果に影響を及ぼすことが考えられます。署名の有効性を検証する時点を、デジタル署名検証ガイドラインではその時刻を「検証基準時刻（validation reference time）」と定義しています。例えば本来の署名検証の目的は署名時点における電子署名の有効性を確認することにあるので、検証基準時刻は“署名を付与した時点”とすることが理想ですが、通常、署名を付与した時刻を客観的に証明することができません。そこで、検証基準時刻はタイムスタンプを併用するなどによる客観的な署名の時刻となり、それが確認できない場合は、署名検証を実

施する現在時刻となります。

実際に証明書の有効性や暗号アルゴリズムの危殆化の有無を判断する際に基準とする検証基準時刻は検証対象により適切に選ぶ必要があります。

証明書の有効性を判断する場合、対象となる証明書についての検証基準時刻は、その証明書をタイムスタンプ対象（MessageImprint）の計算対象に含む有効なタイムスタンプトークンのうち、最も古いものの示す時刻であり、該当するタイムスタンプがない場合、検証処理を実行する時刻となり、検証処理に外部から与える必要があります。

暗号アルゴリズムの非脆弱性を判断する場合、検証基準時刻は、対象となる暗号アルゴリズムにより計算された結果を MessageImprint の計算対象に含む有効なタイムスタンプトークンのうち、最も古いものの示す時刻であり、該当するタイムスタンプがない場合、検証処理を実行する時刻となり、検証処理に外部から与える必要があります。検証基準時刻における検証の考え方を整理すると、以下となります。

・署名

① 署名タイムスタンプがなければ検証時点の現在時刻で検証

② 署名タイムスタンプがあればその時刻で検証

・署名タイムスタンプ

③ アーカイブタイムスタンプがなければ検証時点の現在時刻で検証

④ アーカイブタイムスタンプがあれば最も古いアーカイブタイムスタンプの時刻で検証

・アーカイブタイムスタンプ群

⑤ アーカイブタイムスタンプ群のうち自分より新しいものがなければ、検証時点の現在時刻でそのアーカイブタイムスタンプを検証

アーカイブタイムスタンプ群のうち自分より新しいものがあれば、その直後のアーカイブタイムスタンプの時刻で検証

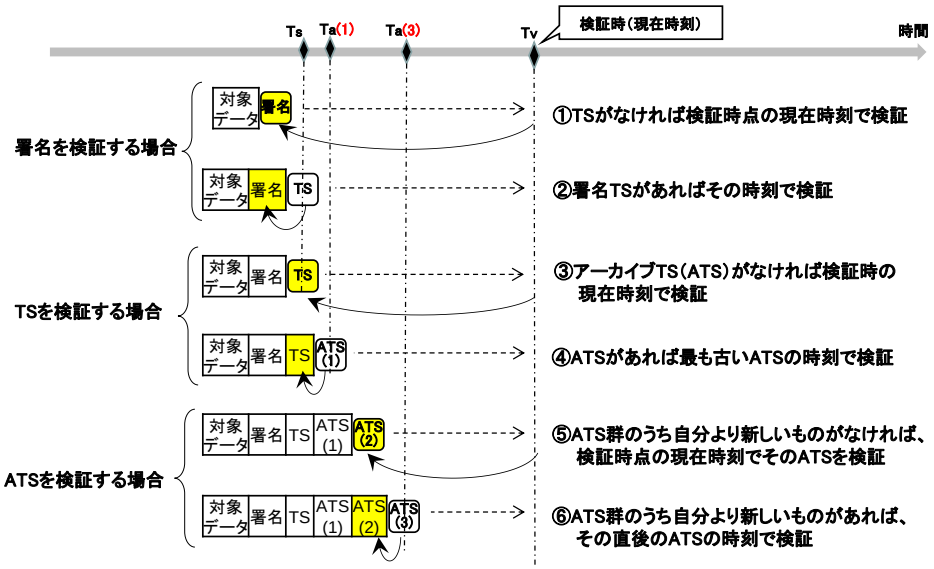

図 4.2.3-1 検証基準時刻の基本的な考え方

長期署名の検証基準時刻は、デジタル署名検証ガイドラインにある通りこれらの組み合わせとなります。

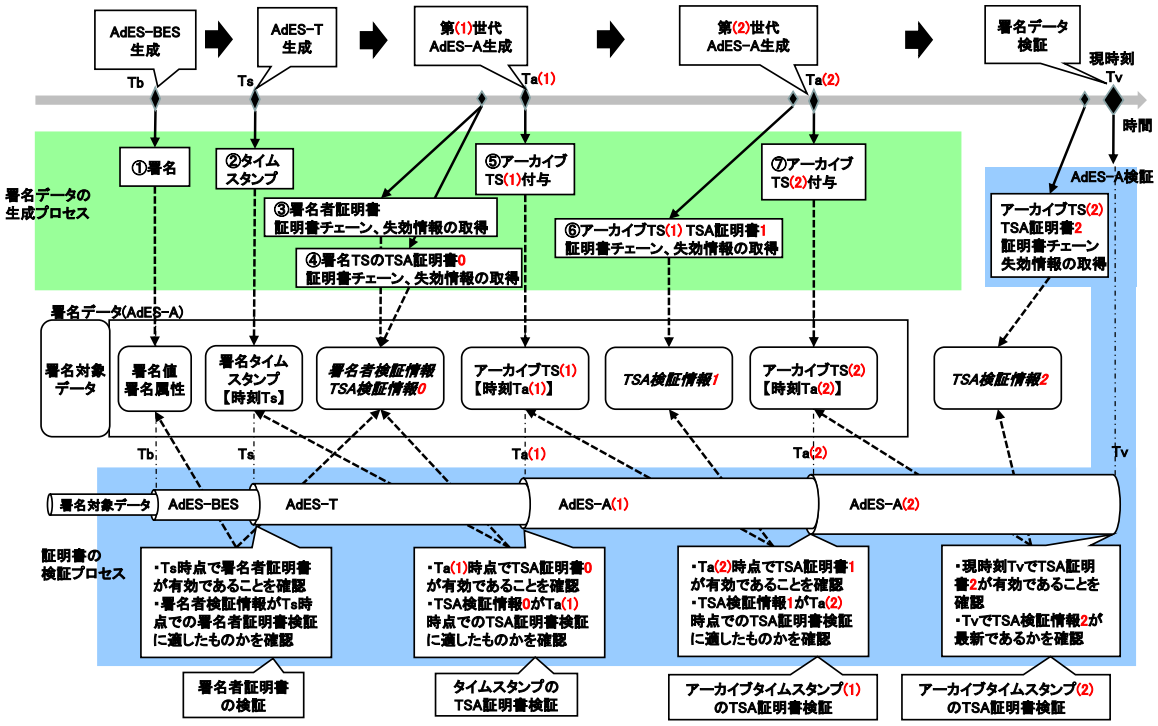

図 4.2.3-2　検証基準時刻の例(アーカイブタイムスタンプ付きの場合)

5 参考文献

本文で引用した参考文献をまとめておきます。

・デジタル署名検証ガイドライン v1.1；JNSA 電子署名 WG
https://www.jnsa.org/result/e-signature/2023/2023-001.pdf
（なお、初版 v1.0 は 2021 年 4 月公開）

・リモート署名ガイドライン；JT2A（日本トラストテクノロジー協議会）
https://www.jnsa.org/result/jt2a/data/RemoteSignatureGguide_All-r1.pdf

・主務三省 Q&A（電子署名法第 3 条関係）に関する解説：JDTF（旧 TSF）および CAC
https://jdtf.or.jp/report/tsf/file/tsf/202102_主務三省 Q%26A_電子署名法第 3 条関係に関する解説.pdf
https://www.c-a-c.jp/download/cmsassets/denshishomei_qa_kaisetsu.pdf

・電子文書長期保存ハンドブック；次世代電子商取引推進協議会,2007.3
https://www.jipdec.or.jp/archives/publications/J0004262.pdf

・PKI 関連技術情報；元は情報処理推進機構のサイトの情報ですが、現在は同サイトから削除のため、国立国会図書館アーカイブを参照
https://warp.ndl.go.jp/collections/info:ndljp/pid/12308150/www.ipa.go.jp/security/pki/

・「暗号技術検討会 2022 年度報告書；暗号技術検討会 CRYPTREC
https://www.cryptrec.go.jp/report/cryptrec-rp-1000-2022.pdf

附録 デジタル署名関連の動向と課題

A.1 署名検証結果のレポート

デジタル署名のフォーマットや検証の判定基準については、デジタル署名検証ガイドラインで示したように標準が定められています。しかし、検証の結果についてはどうでしょうか。検証結果について、時間が経過した後、それが本当にそうであったか、結果自体が改ざんされていないか、あるいは第三者に提示して信用されるためには、検証結果を保証する必要があります。つまり、検証結果にもトラストが必要です。さらに、検証結果は成功／失敗の単純な 2 値ではなく、「不定」の場合や、失敗と判定した理由・根拠を示すことが重要な場合もあり、一種のレポートであると言えます。欧州では、ETSI TS 119 102-2(Signature Validation Report)として標準化されていますので、その概要を紹介します。

A.1.1 欧州の署名検証レポートの規約(ETSI TS 119 102-2)

欧州ではデジタル署名の生成・検証方法の標準化に続き、署名結果のレポート形式の標準化が進められてきました。

表 A.1-1 欧州における署名検証の標準

種別	版番号	題名
Process	ETSI TS 119 102-1 V1.2.1 (2018-08)	Procedures for Creation and Validation of AdES Digital Signatures; Part 1: Creation and Validation
Report	ETSI TS 119 102-2 V1.3.1 (2021-09)	Procedures for Creation and Validation of AdES Digital Signatures; Part 2: Signature Validation Report

A.1.2 署名検証レポートの構造

署名結果の流通性、長期的な保証性を考えると、結果のレポートには必要な項目が含まれ、形式が標準化されていることが重要です。欧州の規約では署名検証レポートは下記の要素で構成され、各々の詳細は XML で規定されています。

・署名検証レポート要素(Signature Validation Report Element)
・署名検証オブジェクト要素(Signature Validation Objects Element)
・検証者情報(Validator Information)
・検証レポート署名(Validation Report Signature)

図 A.1-1　署名検証レポートの構成イメージ（ETSI TS 119 102-2 V1.3.1 より）

（1）署名検証レポート要素情報（Signature Validation Report Element）

この要素は、署名の検証結果情報を表す必須項目です。この要素には、下記が含まれている必要があります。

・Signature Identification Element 署名識別要素
・Signature Validation Status Indication 署名検証ステータスの表示
・Validation Constraints Evaluation Report 検証制約評価レポート
・Signature Validation Time Info 署名検証の時間情報
・Signer's Document Element 署名者のドキュメント要素
・Signature Attribute Element シグネチャ属性要素
・Signer Information Element 署名者情報要素
・Signature Quality Element 署名品質要素
・Signature Validation Process Information Element 署名検証プロセス情報要素
・Associated Validation Report Data Element 関連する検証レポートデータ要素

（2）署名検証オブジェクト要素（Signature Validation Objects Element）

この要素は、検証中に使用される検証オブジェクトの集まりで、オプション項目です。この要素には、検証プロセスで使用された一連の検証オブジェクト要素が含まれている必要があります。署名検証オブジェクトには署名者のドキュメント、トラステッドリスト、失効情報（CRL、OCSP レスポンス）、タイムスタンプなどがあります。

（3）検証者情報（Validator Information）

この要素は、署名を検証し、検証レポートを作成するエンティティに関する情報を含むオプショ

ン項目です。この要素には、欧州の TSP の規約 ETSI TS 119 612 で指定される検証サービスのデジタル ID が含まれている必要があり、その他にも検証者を識別するための追加情報が含まれる場合があります。

（4） 検証レポート署名（Validation Report Signature）

この要素は、検証レポートの署名を含むオプション項目です。この要素には、署名検証レポートの署名が含まれ、検証を実行して検証レポートを作成した検証サービスによって作成されます。

A.1.3 署名検証レポートの今後の展望

署名検証レポートの普及には2つの展開があると考えられます。1つは、デジタル署名が、データの保管や流通を伴う用途で普及し、その時点の結果を証明するニーズが増えることです。短期的あるいは、その場限りの利用、自らが検証して完結する場合などは、レポートにする必要性は必ずしもありませんが、時間経過や第三者への転送を伴う場合は、レポートは有用でしょう。

もう1つのケースは、署名検証を代行するサービスの存在、普及です。署名検証処理は複雑で一般利用者には難しく、検証を代行するニーズがあると考えられます。検証者が検証した結果がその後流通するようなケースでは、その検証結果のトラストが重要となり、結果を保証するレポートとそのフォーマットの標準、さらには、その国際相互運用性が重要となるでしょう。

A.2 有効性モデル(validity model)

デジタル署名を検証するとき、証明書の有効期間と、署名生成時刻及び署名検証時刻との関係に対する捉え方により、有効性の判断に大きく影響を与える 2 つのモデル―有効性モデル(validity model)―があります。1 つはシェルモデルであり、もう 1 つがチェーンモデルです。

証明書は階層型の構成を取り、それぞれが有効期間を持ちます。同じ階層でも様々なタイミングで証明書は発行され、それぞれ異なるタイミングで期限を迎えます。このような場合、証明書の連鎖である認証パスの有効性はどのように考えればよいでしょうか。上位の証明書の保証がある限り有効と考えるならば、最上位(ルート証明書)から入れ子構造になる有効期間が適用されます。一方で発行時に上位の証明書が有効であればよいという考え方もできます。トラストの相互運用性保証のためには、現実に即した考え方と認識の統一が必要です。

以下、それぞれのモデル毎にデジタル署名の有効性の考え方を示します。

A.2.1 シェルモデルにおける有効性

シェルモデルは、日本国内を始め、欧米においても最も一般的に採用されているモデルです。シェルモデルは、RFC5280 に準拠する証明書パス検証アルゴリズムに適合します。

署名者の署名の検証時刻を基準として、その時刻において署名者の証明書からルート CA の証明書に至る認証パス上の証明書のすべてが有効期間内にあることが署名者による署名が有効と判断されるための条件となります(図 A.2-1)。

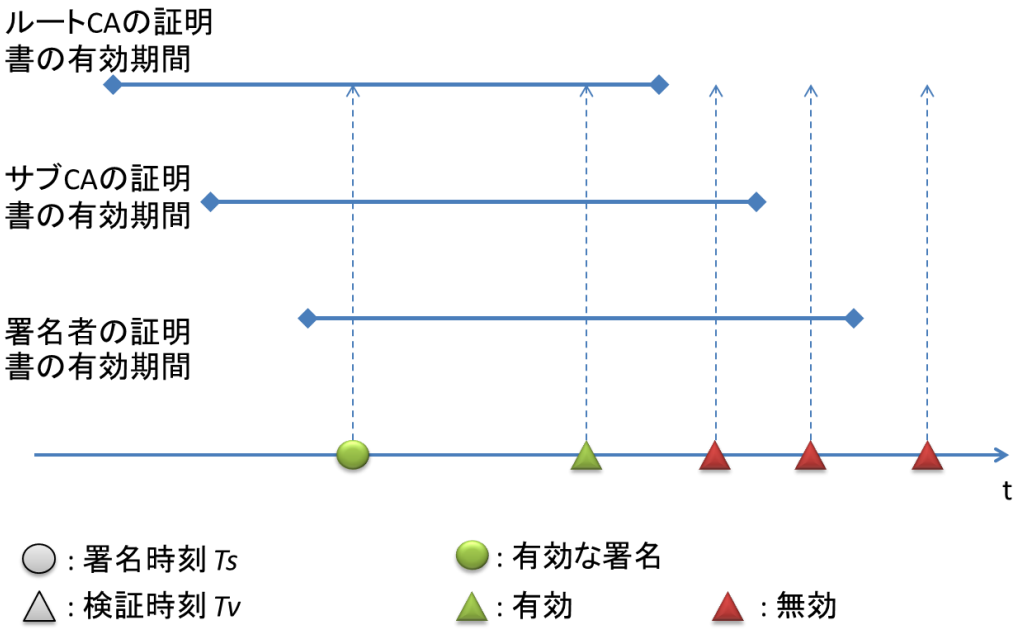

図 A.2-1 シェルモデルにおける署名の有効性

図 A.2-1 では下位の証明書の有効期間が上位証明書の有効期間を超えて設定されているように表しましたが、シェルモデルを前提とする場合、実際には下位証明書の有効期間は上位証明書の有効期間内に含まれるように運用されることがほとんどです。

A.2.2 チェーンモデルにおける有効性

チェーンモデルはシェルモデルと比較すると採用されている場面は少なく、ドイツの QC（適格証明書）による署名の検証の他に、イタリア、ポーランド、エジプトでも利用されているとの情報がありました。（これは数年前の情報であり、現在も利用されていることは確認できていません。）
チェーンモデルは、独の電子署名法からの要請に適合させるために規定された有効性モデルで、ドイツ IT セキュリティ庁が発行した「署名相互運用性仕様」をもとに作成された業界標準である Common PKI[8]の Part 9 SigG-Profile で”SigG model”として説明されています。

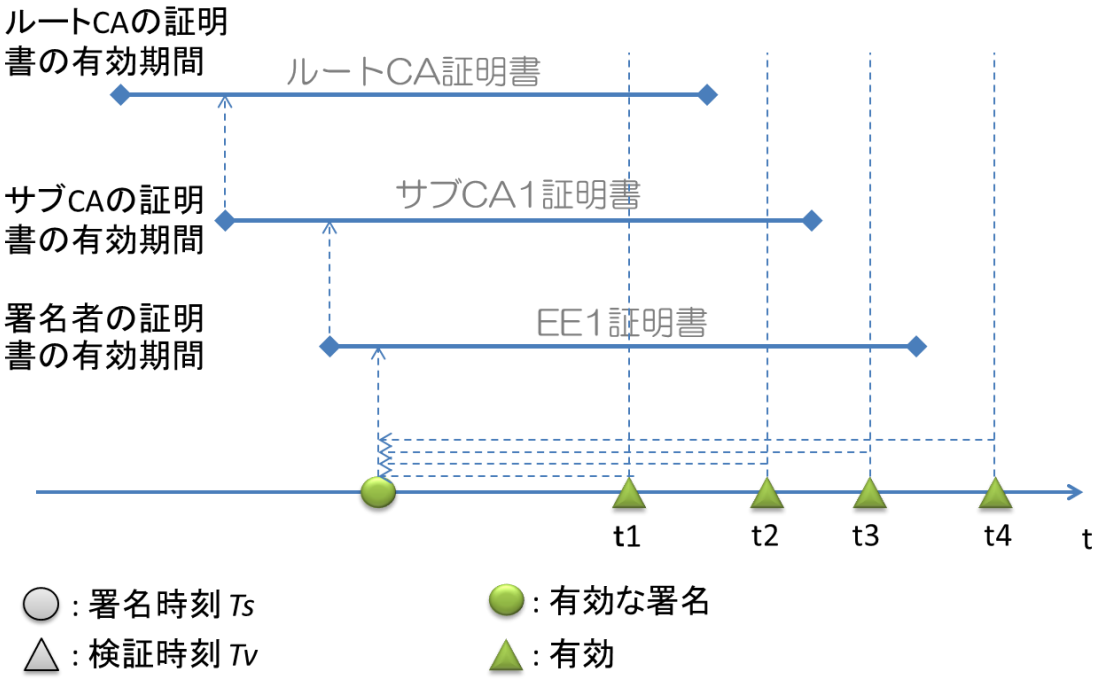

図 A.2-2 チェーンモデルにおける署名の有効性

チェーンモデルでは、図 A.2-2 の t1～t4 のすべてで認証パス上の証明書は（失効していない限り）有効であると判断されます。署名者の署名を始め、各証明書とも上位認証局（ルートについては自己）の署名が付与されており、それぞれの有効性は、署名生成時刻における有効性となります。

[8] Common PKI:Version 2.0 2009/1/20,
https://www.bundesnetzagentur.de/EVD/SharedDocuments/Downloads/QES/Common_PKI_v2.0_02.pdf;jsessionid=3AE14BBFF5CFCCCAB57B46630C5FC211?_blob=publicationFile&v=1

このモデルで検証を実効的なものとするためには、署名生成時刻を証明するための署名タイムスタンプの付与はもちろん、署名生成時、つまり過去における証明書の失効状態を確認する手段が提供される必要があります。そのために、ドイツでは過去の時刻における失効情報の問い合わせが可能な OCSP サービスを提供する場合があります。

A.2.3 検証時の留意点

我が国を始め、多くの国ではシェルモデルによる検証を行っています。「デジタル署名検証ガイドライン」でもシェルモデルによる検証についてのみ記載しています。

本節では、チェーンモデルによる検証の場合をシェルモデルによる検証の場合と比較し、その相違や留意点についてまとめます。

（1） 失効情報の生成時刻の指定の必要性

シェルモデルでは署名者の署名の検証基準時刻は基本的には署名検証を実行する時刻ですが、チェーンモデルでは、検証基準時刻を署名生成時刻とし、その時点で公開鍵証明書の有効性を判断します。従って、署名検証を実行する時刻に発行される失効情報ではなく、署名生成時刻における失効状態を示す失効情報が必要となります。ドイツなどでは、チェーンモデルによる検証の対象となる公開鍵証明書に対しては、過去の時刻を指定してその時刻における失効情報（OCSP レスポンス）を取得できる仕組みを提供しています。ただし、これを実現するためには認証局あるいは検証局において過去に発行した失効情報（CRL など）を保存しておく必要があり、局側の負担は大きくなります。

⑵ 署名タイムスタンプの必要性

チェーンモデルでは、署名の検証基準時刻を署名生成時刻と同時刻とします。このとき、タイムスタンプや認証局の公開鍵証明書の生成時刻はタイムスタンプ局や認証局が保証できますが、署名者の署名時刻を保証するためには署名属性に含まれる署名時刻（SigningTime）ではなく署名タイムスタンプなどの信頼できる時刻保証が必要となります。従って、検証時刻は署名タイムスタンプの時刻としなければなりません。一方でシェルモデルにおいては、署名生成時刻を考慮しないため署名者証明書が有効である間は署名タイムスタンプを必要としません。署名タイムスタンプは署名生成時刻より後の時刻が付与されることとなるため、署名タイムスタンプの取得は署名生成後、可能な限り早くすべきです。

⑶ アーカイブタイムスタンプの必要性

シェルモデルでは、署名タイムスタンプや検証情報の有効期間内にアーカイブタイムスタンプを付与することとなっていますが、チェーンモデルでは、署名者の署名で利用されるハッシュ関数、署名タイムスタンプで署名対象からメッセージインプリントを得る際に利用されるハッシュ関数が危殆化しなければアーカイブタイムスタンプを付与する必要がなく、シェルモデルの場合と

比較してアーカイブタイムスタンプを取得する頻度が下げられます。

上記より、チェーンモデルを採用する場合、認証局や検証局の負荷（失効情報の長期にわたる保持）は高まるというデメリットはありますが、利用者にとってはタイムスタンプ取得の頻度を下げられるというメリットは得られます。シェルモデルではチェーンモデルとメリット／デメリットが逆となります。

A.3 各種データ形式

デジタル署名検証ガイドラインで示したデジタル署名のデータ形式は、発生の経緯と普及の度合いから、CAdES、XAdES、PAdES について記載しています。しかし、IT 技術の進展、各種データ形式利用環境の制約などから、新たな形式の出現や、実用上の課題が生じます。ここでは、新たな形式 JAdES と、PAdES の実用上の留意点について述べます。

A.3.1 長期署名基本フォーマット CAdES/XAdES

まず長期署名の基本となる CAdES と XAdES を確認します。この 2 つはベースフォーマットと署名対象指定を除けばほぼ同じ仕様であり、長期署名の基本フォーマットと言えます。

表 A.3-1 長期署名の基本フォーマット

	CAdES	XAdES
ベースフォーマット	CMS（PKCS#7）、ASN.1/BER	XML 署名（XML-DSig）、XML
署名対象指定	基本1つ（複数対象の ZIP 化は可） Attached/Detached	URI で複数指定可能 Enveloping/Detached/Enveloped

A.3.2 PDF ドキュメント長期署名 PAdES

PAdES はベースフォーマットとして PDF 署名を利用していますが、PDF はドキュメントフォーマットである点が CAdES/XAdES とは異なります。とは言え PDF 署名にて埋め込まれている署名データは PKCS#7 形式でした。その意味では PKCS#7 を拡張した CAdES を採用すれば簡単に長期署名化が簡単でしたが、署名データの埋め込みサイズは署名時に ByteRange 要素で決められてしまい検証情報の埋め込みが出来ない問題がありました。そこで PAdES では CAdES-T（署名タイムスタンプ付）までを採用し、検証情報は PDF 構造の中に DSS/VRI 辞書として埋め込むことになりました。またタイムスタンプも新たに DocTimeStamp と言うタイムスタンプ単体利用を可能としました。長期署名以前のPDF署名仕様でも利用可能でしたが署名外観（印影等）が使えると言う点もPDF署名の利点と言えます。ただし署名外観は明確に ISO 仕様がある訳ではないデファクト標準となっています。この為に署名外観を表示できないPDF表示アプリケーションもあります（通常その場合は PDF 署名そのものに未対応であることが多い）。

表 A.3-2 PAdES 仕様の概要

要素	PAdES 仕様
署名データ	CAdES-BES（署名のみ） / CAdES-T（署名＋タイムスタンプ） ※ Adobe 独自の失効情報埋め込み仕様あり
検証情報埋め込み	DSS/VRI 辞書 （PDF への証明書/CRL/OCSP の埋め込み）
タイムスタンプ（単独）	DocTimeStamp （タイムスタンプのみの利用が可能）

署名外観(印影等)	標準化された仕様では無いが画像等の署名外観が設定可能

つまり PAdES は CAdES をベースにしていますが、検証情報埋め込みとタイムスタンプ付与に関しては独自仕様を持つことになりました。加えて PDF 署名は順番に署名を追加していくシリアル署名方式のみが利用可能である点が大きな特長かつ制限となっています。署名データ・タイムスタンプ・検証情報の 3 つ任意の順番に重ねて行くことができるようになりました。この影響により PAdES では ES-BES→ES-T→ES-X-Long→ES-A と言うサイクルに縛られない運用も可能となりました。これは CAdES/XAdES では出来なかったタイムスタンプのみの運用を可能とした一方で、複数署名の長期署名においてどう運用して良いか不明瞭になる問題を生じました。PDF 仕様である ISO 32000 では PAdES 署名の仕様は分かりますが、長期署名の運用については記載がありません。長期署名の運用を補う仕様が ISO 14533-3 となっています。また PAdES の運用に関してはデジタル署名検証ガイドラインの「付属書 B (参考): PAdES 関連情報」としてまとめてありますので一読願います。

ドキュメントフォーマットとしては PDF 以外であれば、Office 文書の仕様である OOXML:Office Open XML (ISO/IEC 29500) があります。OOXML は XML ベースである為に XML 署名を採用していましたが、これを単純に XAdES に変更するだけで長期署名になりました(PDF の ByteRange のような問題がありませんでした)。OOXML には OPC(Open Packaging Conventions)と言う ZIP 化コンテナ仕様もありますので、ドキュメントフォーマット以外にコンテナフォーマットしての利用も可能となります。

表 A.3-3 ドキュメントフォーマットの長期署名化

	PDF(PDF 署名)	OOXML(ZIP化＋XML 署名)
長期署名化	PAdES 仕様:CAdES-T と PDF への検証情報埋め込み	XML 署名の XAdES 化のみで良い Manifest で署名対象を指定
長期署名仕様	ISO 32000 ＋ ISO 14533-3	ISO/IEC 29500 ＋ ISO 14533-2
複数署名方式	シリアル署名	パラレル署名
署名外観	あり	なし(署名パネルはある)

PAdES 検証器として Adobe 社の Acrobat Reader が普及している点は PAdES 利用の大いなる利点と言えます。Adobe では LTV(Long-Term Validation)と言う仕様を利用しています。署名パネルにおいて「署名は LTV 対応です」と表示されます。LTV 状態は長期署名で言えば検証情報を埋め込んだ PAdES-X-Long と言えます。このように Adobe 製品の長期署名表示は ISO 14533-3 等とも少し異なっています。また検証仕様もデジタル署名検証ガイドラインと検証項目が 100%一致しているわけではありません。Adobe の検証にはこのような問題はありますが差異を理解してうまく使うことで、デファクト検証器として利用が可能です。

A.3.3 JSON データ長期署名 JAdES

WebAPI においては長く XML が使われて来ていましたが、近年ではよりシンプルな JSON の利用が大半を占めるようになりました。現在では WebAPI のうち RESTful な API は JSON を使っていると言えます。また Web サービス間の連携時にはデジタル署名を付与する必要があり JWS（JSON Web Signature/RFC 7515）仕様が一般的に使われているます JWS にはトークンとして利用する為の JWS Compact Serialization 仕様（JWT/RFC 7519）と、フル仕様である JWS JSON Serialization 仕様の 2 つがあります。JWT は Web サービス間にてトークンとして利用される署名フォーマットであり長期署名化のニーズはほとんどありません。一方で JWS JSON Serialization 仕様を利用した長期署名フォーマット JAdES（ETSI TS 119 182）が存在します。JAdES はまだ ISO や RFC の標準化はされていません。仕様的には XAdES に近い仕様となっています。

```
{
  "payload":"<payload contents>",
  "protected":"<integrity-protected header contents>",
  "header":<non-integrity-protected header contents>,
  "signature":"<signature contents>"
}
```

図 A.3-1　フラット化されたJWS　JSON Serialization の例

表 A.3-3　JAdES と XAdES の構造比較

要素	JAdES/JWS	XAdES/XML-DSig	補足
署名対象	Payload	SingedInfo	XML-DSig では署名対象を URI として間接指定すると共に署名アルゴリズム等を指定
署名値	Signature	SignatureValue	どちらも Base64 化して格納
署名対象属性	Protected	SignedProperties	署名時刻や署名証明書等、JWS では署名アルゴリズム等を指定
非署名対象属性	Header	UnsignedProperties	タイムスタンプや検証情報等の追加要素

XML 署名/XAdES では XML データに署名する為に XML-C14N（XML Canonicalization）と呼ばれる正規化仕様を利用しています。一方 JWS（JSON 署名）では署名対象をバイナリデータとして扱い Base64 化することで正規化を行わないで済むようにしています。XML-C14N 仕様は比較的難解であり正規化処理から解放されたと言う意味では JWS/JAdES は実装が容易になったと言えます。一方で長期署名フォーマットである JAdES の場合には検証情報や署名対象を何度も Base64 化する必要があり JAdES データの可読性と言う意味ではやや問題を持っているように思

えます。

JWS では署名対象は1つのみであり CMS/CAdES に近い仕様と言えます。JAdES ではこれを補う為に URI 参照(URI Reference)の仕様を追加しています。ただし URI 参照は JAdES 独自の仕様であり、ベースとなっている JWS の仕様には含まれていない点には注意が必要となります。つまり JWS としての検証だけでは検証できないと言うことです。

以上 JAdES を使う場合の注意点としては、多重 Base64 化による可読性低下の問題と、複数の署名対象を URI 参照して使う場合の JWS 非互換の問題、があることになります。それらを除けば XML の長期署名である XAdES とほぼ互換と言えます。長期署名の基本フォーマットはこれで CAdES/XAdES/JAdES の 3 種類に増えました。それぞれの特長や問題を理解した上で使い分けることができます。

表 A.3-4 JAdES と XAdES の比較

	JAdES	XadES
ベースフォーマット	JWS(JSON 署名)、JSON	XML 署名(XML-DSig)、XML
署名対象指定	基本1つだが JAdES 独自拡張の URI 参照により複数指定可能	URI で複数指定可能 Enveloping/Detached/Enveloped
署名対象の扱い	バイナリとして Base64 化	XML-C14N 正規化が必要(XML 以外はバイナリ指定も可能)

A.3.4 長期署名フォーマットまとめ

長期署名系のフォーマットとしては他にも ZIP 長期署名コンテナ ASiC(ETSI EN 319 162)もあります。これはドキュメントフォーマットの長期署名化に近いですが、欧州が独自に(ISO や RFC 化されていない)ZIP 圧縮ファイルの長期署名化を行ったものであり、長期署名基本フォーマットしては XAdES/CAdES を利用しています。

ここまで解説した通り長期署名フォーマットしては大別すると基本フォーマットとドキュメントやコンテナを長期署名化する応用フォーマットの 2 種類があります。利用目的に合った長期署名フォーマットを利用することになります。

表 A.3-5 長期署名フォーマットの種類

大別	名称	概説
基本	CAdES	CMS(ASN.1/BER 署名)の長期署名基本フォーマット 仕様: ISO 14533-1, RFC 5652, ETSI EN 319 122, ETSI TS 101 733 ※ ASN.1/BER はバイナリなので署名データが小さくなります
	XAdES	XML 署名による長期署名基本フォーマット 仕様: ISO 14533-2, W3C XML-DSig, ETSI EN 319 132, ETSI TS 101

		903 ※ 署名対象の指定が多様であり可読性も高いが XML 正規化が必要
	JAdES	JWS（JSON 署名）による長期署名基本フォーマット 仕様: RFC 7515, ETSI TS 119 182 実装が比較的容易で JWS/JWT 利用は拡大中 ※ 長期署名時は多重 Base64 化による可読性と URI 参照に注意が必要
応用	PAdES	PDF の長期署名ドキュメントフォーマット 基本: CAdES-T を利用し検証情報埋め込みのみ PAdES 独自 仕様: ISO 14533-3, ISO 32000-2, ETSI EN 319 142, ETSI TS 102 778 ※ シリアル署名方式であり長期署名や複数署名の運用には注意が必要
	ASiC	ZIP 化による長期署名コンテナフォーマット（欧州独自） 基本: XAdES か CAdES を利用 仕様: ETSI EN 319 162, ETSI TS 102 918 ※ ZIP 化コンテナ長期署名は OOXML/OPC（ISO/IEC 29500）等もあります

A.4 トラストフレームワークと国際相互承認

グローバル化するビジネス環境において、国境を超えるデータは、国際的な相互運用において信頼関係を保証することが必須となります。国内外のトラストのフレームワークは官民合わせていくつかありますが、国際的な相互認証についてはまだ議論が始まったところです。その考え方と動向について紹介します。

A.4.1 データのトラストの国際相互運用

ビジネスや社会課題の解決に有益なデータがプライバシーやセキュリティ、知的財産権に関する信頼を確保しながら国境を意識することなく自由に行き来する、というコンセプト、DFFT（Data Free Flow with Trust：信頼性のある自由なデータ流通）が、2019 年 1 月にスイス・ジュネーブで開催された世界経済フォーラム年次総会（ダボス会議）にて、日本から提唱されました。

2023 年 4 月に開催された「G7 群馬高崎デジタル・技術大臣会合」においても DFFT は改めて注目され、DFFT の具体化のための国際枠組み（IAP：Institutional Arrangement for Partnership）の設立が合意されました。

国際的に DFFT を実現するためには、データ流通層やトラストサービス層の国際相互運用が求められ、特にトラストサービス層では、相手国の電子署名やタイムスタンプなどが国境を越えて相互にその信頼性を検証できることが必要となります。

A.4.2 国内外のトラストフレームワークとトラストアンカー

トラストを確保する枠組み、トラストフレームワークは例えば法人向けオンラインバンキングなど企業やある特定のサービスに閉じて実現するプライベートなものや、電子署名法の認定認証業務やインターネットサイトで用いられる Web サイト証明書のように不特定のサービスで広く用いられるパブリックなトラストフレームワークに大別できると考えられます。ここでは、代表的なパブリックなトラストフレームワークを紹介します。

（1）Web サイト証明書のトラストフレームワーク

Web サイトが、その運営主体が主張する通りの本物のサイトであることを、アクセスした利用者が検証できるようにするため、第三者としての認証局が Web サーバー運営主体の正当性を確認して Web サイト証明書を発行する仕組みがインターネットの信頼性を確保するために用いられています。Web サイト証明書により、サイトの真正性が確認でき、当該サイトと利用者のブラウザー間でセキュアチャンネルが確立され、経路を流れる通信データを暗号化することによりパスワードやクレジットカード情報などプライバシーや機密にかかわる情報漏洩を防止し安心、安全にインターネットを利用することを可能としています。

この Web サイト証明書のトラストフレームワークを図 A.4-1 に示します。

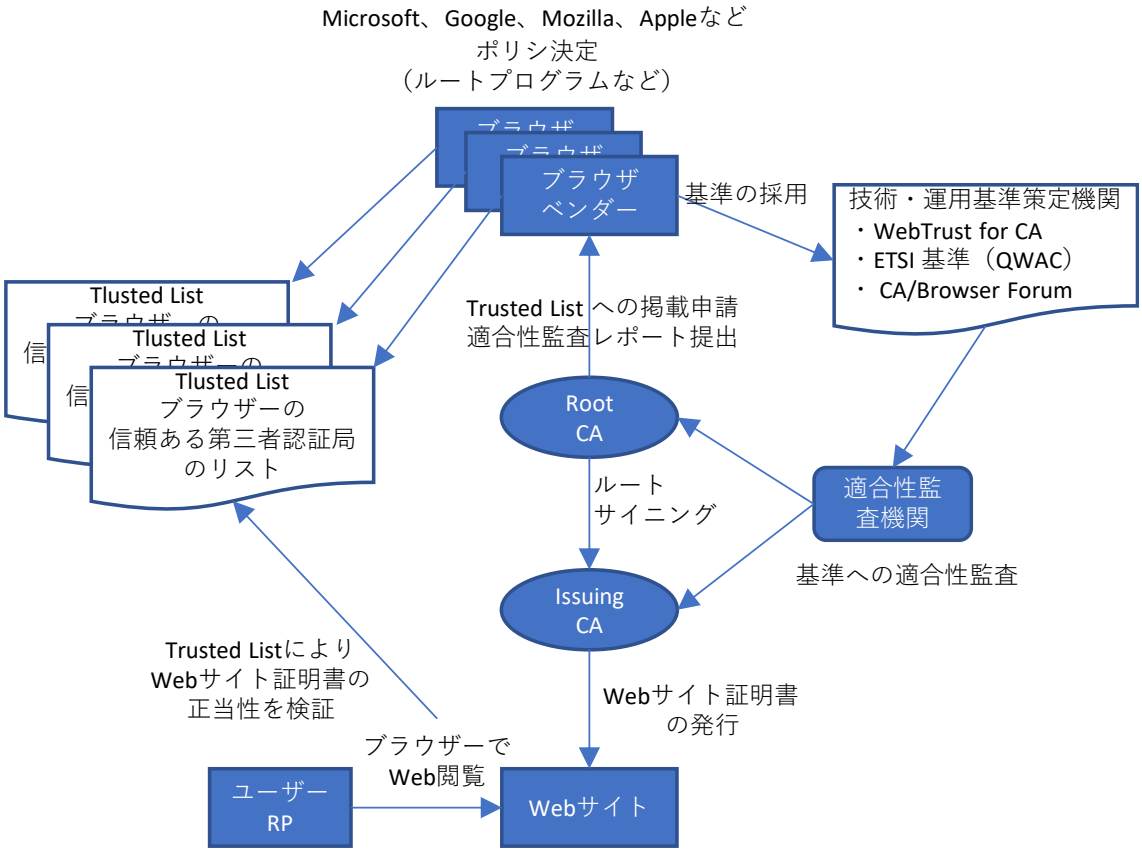

図 A.4-1　Web サイト証明書のトラストフレームワーク

Web サイト証明書の信頼性を検証できるようにするため、各ブラウザーベンダーは自社のブラウザーの中に信頼性が確認されたルート証明書のリストを保持しています。このトラストフレームワークでは各ブラウザーベンダーが、WebTrust や CA/Browser forum が定めた認証局の技術・運営基準への準拠性監査の結果を基に自社のブラウザーのリストへ掲載するルート CA 証明書としています。

(2) わが国の政府ブリッジ認証局を中心としたトラストフレームワーク

わが国の電子政府を支える基盤として、インターネットを利用した申請・届出等において、行政機関からの電子的な通知等や、利用者からの電子申請データの真正性（データが本当にその名義人が作成したものであり改ざんされていないこと）を確認するための仕組み、政府認証基盤（GPKI：Government Public Key Infrastructure）が整備されています。

GPKI はブリッジ認証局（BCA：Bridge Certification Authority）と政府共用認証局等から構成され、図 A.4-2 のように BCA を中心に相互認証を行うことにより BCA と繋がれた認証局の証明書を相互に検証可能としています（ブリッジモデル）。電子申請等の利用者に証明書を認証する民

間認証局のうち、電子署名法の特定認証業の認定を取得した認定認証局は、BCA と相互認証が可能となります。(電子申請の利用者向けに証明書を発行しない認定認証局は BCA と相互認証しない場合もあります)。

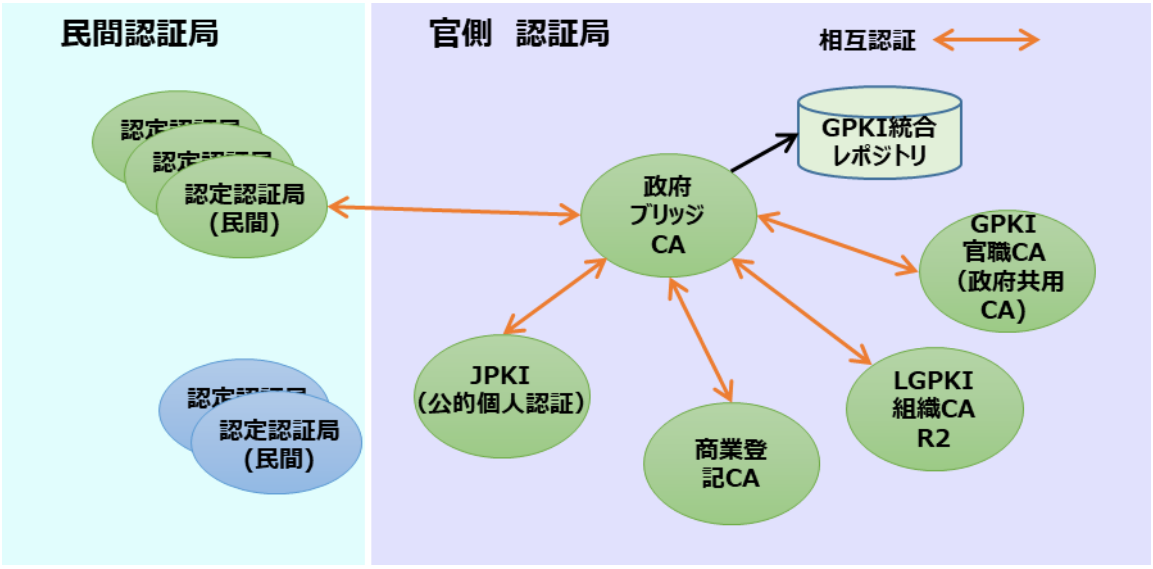

図 A.4-2　政府認証基盤(GPKI)と電子署名法の認定認証局

BCA を介したブリッジモデルによる証明書検証は、電子申請を行う省庁側では検証サービスが用意されているため容易に検証可能となりますが、一方、民間分野でブリッジモデルによる検証を提供しているサービスは実際には存在しないため、認定認証局から発行された証明書を検証するには、署名法の認定を受けた認証局であることを個別に確認する必要があります。

電子署名法に基づくトラストフレームワークを図 A.4-3 に示します。認定の基準は電子署名法の施行規則や認定の指針等で定められ、指定調査機関が行う監査の結果を受けて主務大臣が認定します。認定された認定認証局は官報に CA 証明書のフィンガープリントが掲載されるとともに、主務省のホームページで認定認証業務の名前が公開されます。

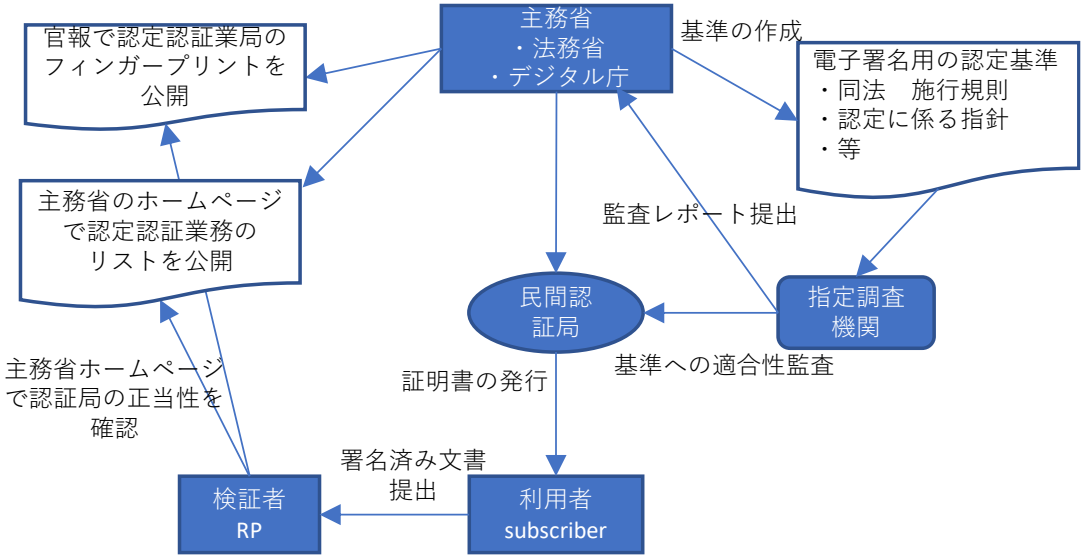

図 A.4-3　電子署名法に基づくトラストフレームワーク

(3) Adobe 社の AATL(Adobe Approved Trust List)のトラストフレームワーク

電子署名が付与された PDF 文書(PAdES フォーマット)を Adobe Acrobat など Adobe 社の製品で検証する際には、同社の製品に組み込まれている AATL(Adobe Approved Trust List)により証明書の正当性を確認します。AATL のトラストフレームワークを図 A.4-4 に示します。

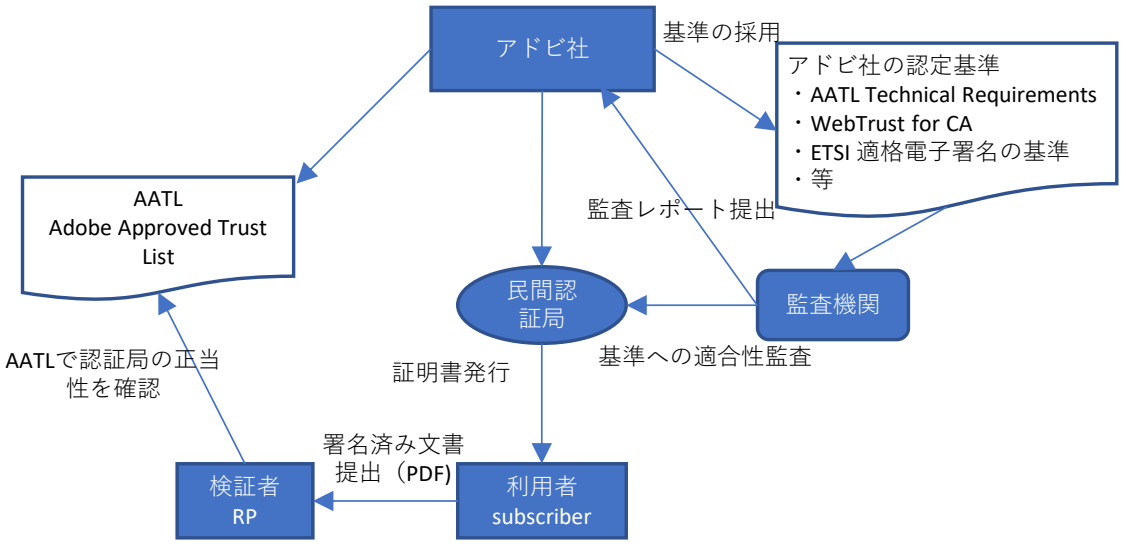

図 A.4-4　AATL のトラストフレームワーク

認定基準は、Adobe 社が定める、AATL Technical Requirements に加え、WebTrust for CA または ETSI 適格電子署名の基準です。監査機関により準拠性が確認された認証局の証明書が AATL に掲載されます。

(4)欧州における eIDAS 規則に基づくトラストフレームワーク

欧州では、eIDAS(Electronic Identification, Authentication and Trust Services)規則に認証局などのトラストサービスプロバイダーの認定スキームが定められており、図 A.4-5 のトラストフレームワークが構成されています。認定基準は欧州委員会のイニシアティブにより ETSI などの標準化機関が認証局などの TSP の認定基準や適合性評価機関の要件を定めています。適合性評価機関の監査レポートにより各国の国家監督機関は適格トラストサービスを認定し、各国の Trusted List に当該適格 TSP の証明書を掲載します。欧州委員会は各加盟国の Trusted List の一覧表(URL のリンク情報)を掲載した List of Trusted List を発行し、適格証明書の有効性確認は EU LOTL がトラストアンカーとなり一元的に確認できるようにしています。

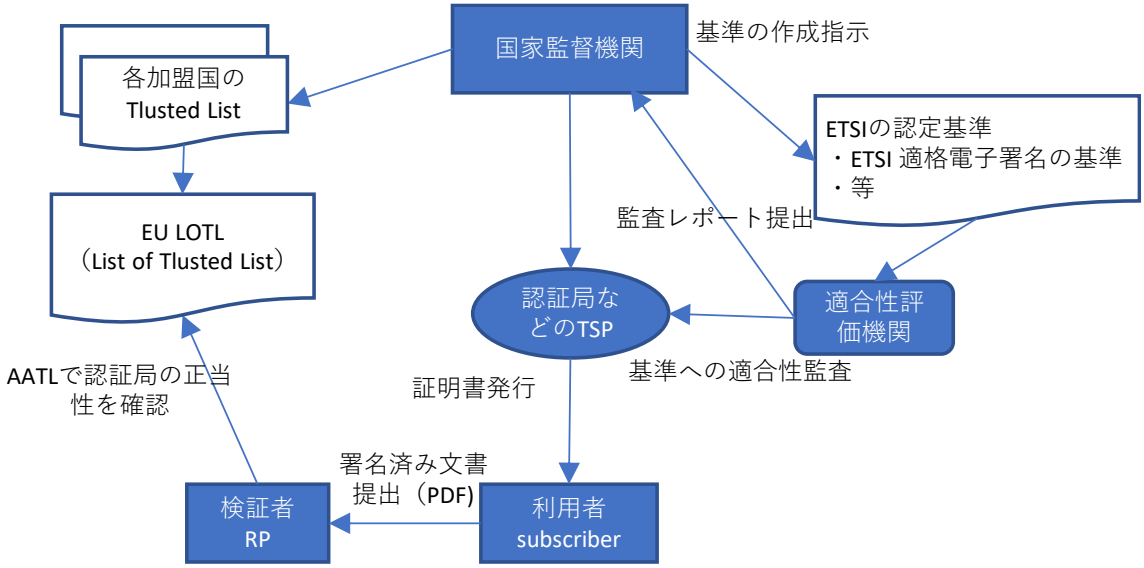

図 A.4-5 EU の eIDAS 規則に基づくトラストフレームワーク

A.4.3 トラストアンカーの相互接続

前節では、様々なトラストフレームワークを見てきましたが、トラストフレームワークを構成する要素は

(1) 制度を統括するトラストフレームワークの提供者
(2) 認定基準
(3) 監査スキーム
(4) トラストアンカーの開示の仕組み

証明書検証の原点は、トラストアンカーを確認するところから始まります。それは認定基準に基づき、監査機関により認証局の監査が行われ、トラストフレームワーク提供者が認定し、トラストアンカーを開示する一連のトラストフレームワークのプロセスにより支えられています。

トラストサービスの国際的な相互認証を実現するには、国を超えて異なるトラストフレームワーク間で相互に相手のトラストフレームワークを承認する国際相互承認（MRA:Mutual Recognition Agreement）を締結する必要があります。

令和 3 年4月にデータ戦略タスクフォースの下に設置され、内閣官房 情報通信技術（IT）総合戦略室にて開催された「トラストに関するワーキングチーム」では、国際的な相互承認を得るためには、「国際間の利用者が相互に適格性を確認できるように、以下の項目の同等性などを検討し、相違点を補完する仕組みが必要ではないか。」とされ、以下の 4 つの観点での同等性の確認が必要であることが示されています。

1. 法制度（法的効果の同等性）
2. 監督・適合性評価
3. 技術標準(可能な範囲で技術規格をそろえるが、必ずしも同一である必要はない)
4. トラストアンカーの確認、トラストアンカー間の接続の仕組み

例えば、欧州と日本の間で、MRA を締結し相手方の電子署名を相互に検証可能とするためには、欧州のトラストアンカーと日本のトラストアンカーをお互いに受け入れる必要があります。検証者（relying party）が個々に手動で相手先のトラストアンカーを確認するのはあまりにも煩雑であり、電子署名の相互検証を機械的に可能とするために、それぞれのトラストフレームワークにおいて機械可読な形でトラストアンカーを開示した上で、相手側のトラストアンカーと接続することが必要となります。図 A.4-6 に SIP[9]で技術実証された、欧州と日本のトラストアンカーの接続の仕組みを示します。

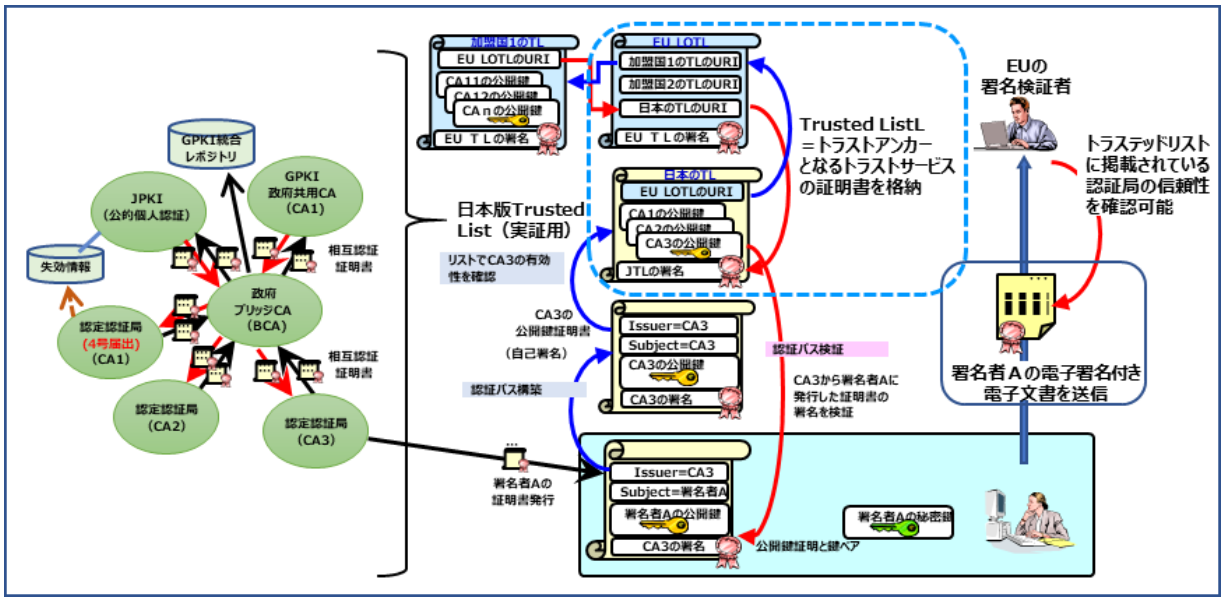

図 A.4-6 トラストアンカーの相互接続によるトラストサービスの国際相互運用

[9]「戦略的イノベーション創造プログラム（SIP）第2期／ビッグデータ・AIを活用したサイバー空間基盤技術／分野間データ連携基盤技術／分野・組織を超えたデータ活用とサービス提供を実現する基盤の研究」「国際的な相互連携実現に向けた取り組み/トラスト基盤連携」（内閣府）

この技術実証では、政府 BCA と接続する認証局を、実証用の日本のトラステッドリスト（JTL）に掲載し、実証用の EU の List Of Trusted List に相互参照ポイントとなる URL などを記載することで電子証明書が相互に検証可能となり相互運用が実現できることが実証されています。

A.5 リモート署名とｅシール

従来、デジタル署名はハンコのイメージで、主に自然人が、手元で実施するものと想定されてきました。その一方で、署名者の証明や改ざん検知機能の利便性から、サーバー証明書や役職証明書、コード署名やタイムスタンプなど、必ずしも自然人が手元で行うとは限らないものも普及してきました。昨今、これらの状況を再整理し、用途や形態に応じて適切に基準を策定する流れがあります。ここでは、利用者の手元で処理しないリモート署名と、自然人に限らない ｅ シールについて紹介します。

A.5.1 リモート署名

リモート署名は、電子署名及び認証業務に関する法律（平成 12 年法律第 102 号）に基づく電子証明書の普及と制度の企画をデジタル庁が一体的に担うことになっています[10]。リモート署名は、利便性を向上させる一方で、どのような要件を満たせば、本人による電子署名といえるのか、制度的な整理がまだ明確ではありません。リモート環境へのアクセス方法等について、本人だけが行うことができるといえるだけの技術的要件を明確化する必要があります[11]。この点については、現在、日本トラストテクノロジー協議会（JT2A）により、EU におけるリモート署名関連の標準を参照しながら、リモート署名の技術的要件を検証した結果がリモート署名ガイドラインとして公表されています。

リモート署名とは、利用者の署名鍵を事業者のサーバーに保管し、利用者がサーバーにリモートでログインして電子署名を行うものです。そのため、署名鍵を自己管理する必要がなく、いつでもどこでも電子署名が可能ですが、一方で、署名の改ざんやなりすましを防止する機能が必要になります。例えば、利用者認証機能、利用者管理機能、署名機能、署名生成ログ機能、署名鍵管理機能、署名鍵バックアップ機能から構成されます。オンライン上の攻撃やすり替えに対して対策する必要があり、署名アルゴリズムや鍵長は安全なものを選択する必要があります。また、クラウド上でサービスを提供する場合もあり、コストや運用負荷を削減できる場合もあります。

リモート署名の制度化に向けた論点については、法的な問題や認証制度の整備、セキュリティ要件や技術要件などが挙げられます。電子署名は、平成 13 年 4 月から施行された「電子署名法」に基づいて行われるものであり、リモートに関する法的な問題が未整理のため、制度化に向けた論点として注目されています。また、リモート署名においては、セキュリティ要件や技術要件が重要視されます。具体的には、署名鍵の保管方法や認証局の運用方法などが挙げられます。さらに、リモート署名においては、認証制度の整備が重要です。これらの認証制度を整備することで、電子署名の信頼性を高めることができます。

[10] 総務省｜令和 4 年版 情報通信白書｜デジタル庁における検討状況
https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r04/html/nd245440.html
[11] リモート署名の制度化に向けた論点について
https://www.soumu.go.jp/main_content/000654276.pdf

A.5.2 e シール

e シールとは、法人や組織の電子的な印鑑や署名を表す技術です。電子署名は個人の身分証明や否認防止に重点を置いていますが、e シールは法人や組織の存在証明や信頼性の向上に重点を置いています。e シールが求められる背景としては、電子文書の増加や法人間の取引の高度化、国際的な相互運用性の確保などが挙げられます。さらに今後、e シールを普及させる必要がある理由としては、法人や組織の業務効率化やコスト削減、セキュリティやコンプライアンスの強化、デジタルトランスフォーメーションの推進などが挙げられます。また、e シールはデジタル社会の推進においても重要な役割を果たします。例えば、電子契約や電子決済、電子証明書や電子公文書などの分野で、法人や組織の信頼性や信用性を高めることができます。

e シールの制度化に向けた論点としては、法的効力や証明力、国際的な相互運用性、利用者の利便性や安全性、認証制度の整備などが挙げられます。e シールのセキュリティ要件や技術要件の重要性は、e シールが偽造されたり改ざんされたりしないようにすることで、法的効力や信頼性を確保するためです。e シールの認証制度の重要性は、e シール証明書を発行する認証局が信頼できるものであることを保証することで、e シールの信頼性や信用性を高めるためです。また、認証局によって発行された e シール証明書は、国内外で相互運用可能であることが望まれます。

A.5.3 リモート署名と e シールに関する署名検証の考慮点について

(1) リモート署名における考慮点

前述のリモート署名と検証の関係として、検証時に検証できることが望ましい項目があります。例えば、ローカル署名ではなくリモート署名によって署名されたことが検証できることです。一般的な署名では署名者のローカル環境で署名するため、署名する環境や署名鍵の保存については安全性を問われることが少ないですが、リモート署名では、リモート(遠隔)であるため署名する環境や署名鍵の保管に、より安全性が問われる懸念があります。そのため、リモート署名で利用する署名鍵の生成場所や署名鍵を生成した HSM からの移動はないこと、保管場所を明確にすることで、署名検証時に確認することができます。詳細については、JT2A のリモート署名ガイドライン パートⅢの付録 1 の「8 署名鍵の生成環境の区別」に記載しているので参照してください。

(2) e シールにおける考慮点

eシールは、発出元が自然人ではなく、組織であることから意思表示がありません。そのため、自然人による電子署名と区別して確認できることが求められます。総務省の検討[12]においても e

[12] 組織が発行するデータの信頼性を確保する制度に関する検討会取りまとめ(案)
https://www.soumu.go.jp/main_content/000744787.pdf

シールは、データ発行元の組織を簡便に確認できるようになるとともに、経理関係業務等のデータに対して機械的に迅速・大量にｅシールを付すため、自然人による電子署名とｅシールの利用シーンは異なることが想定されており、電子署名ではなくｅシールであることを検証できることが望まれます。

A.6 証明書の適用領域による検証時の留意点

証明書は適用領域や利用用途に応じて、検証方法に注意すべき点があります。標準仕様である ITU-T X.509[13] や RFC 5280[14] の 6 章では証明書の認証パス検証のアルゴリズムの例について規定されていますが、文書署名用の証明書か、タイムスタンプ用の証明書かによって追加の検証が必要になりますし、業界、業種、アプリケーションもしくは法制度の定めにより検証要件が追加されたり、また、認証局によって失効情報の提供に違いがあるために、これに配慮する必要があるケースがあります。本節では、そのような証明書検証の留意点について述べます。

A.6.1 一般的な利用用途による追加検証

証明書と秘密鍵さえあれば、文書やタイムスタンプに署名できてしまうアプリケーションはあり、例えば TLS サーバー署名証明書、クライアント認証用の証明書、コード署名証明書であっても文書に署名できてしまうことがあります。そのため、証明書が文書署名用なのか、タイムスタンプ専用なのかは確実に検証する必要があります。具体的には以下の拡張領域を追加検証することにより証明書の用途を検証します。

- 基本制約 CA フラグ(basicConstraints)がないこと
- 鍵使用目的 (digitalSignature, nonRepudiation/contentCommitment)
- 拡張鍵使用目的 (timeStamping, documentSigning)

注 1: ITU-T X.509 10/2019 より nonRepudiation は contentCommitment になりました。

注 2:RFC 9336[15] より documentSigning の拡張鍵使用目的が使えるようになりました。

A.6.2 信頼するルート認証局/発行用認証局のリストの違い

検証しようとしている署名やタイムスタンプの証明書について、どの認証局を信頼点(のリスト)として検証すべきか、そのアプリケーション領域、ドメインごとに指定されているはずです。検証された証明書チェーンについてルート CA 証明書や中間 CA 証明書が信頼しているリストに含まれているのか確認する必要があります。

具体例として以下の信頼点(のリスト)が提供されることがあります。

① OS がデフォルトで持っている信頼する認証局のリスト(Windows, Android, iOS, macOS, Linux 等)

② アプリケーションが持っている信頼する認証局のリスト(Adobe Acrobat AATL, OpenSSL, Mozilla 等)

③ 日本の GPKI の指定された信頼点(府省 CA 等)

[13] ITU-T X.509 (10/2019) Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks

[14] RFC 5280 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile

[15] RFC 9336 X.509 Certificate General-Purpose Extended Key Usage (EKU) for Document Signing

④ 米国ブリッジ CA につながる信頼点として指定された CA
⑤ EU トラストリスト
⑥ 医療等、業界、業種に限定される信頼点リスト

A.6.3 業界やアプリケーションに依存する証明書ポリシーの検証

(1) 証明書ポリシーの基本的な検証

RFC 5280 の 6 章には証明書チェーンの認証パス検証のアルゴリズムの例が規定されていて、その中には証明書ポリシーのあるべき処理が含まれています。アルゴリズムが複雑で、6 章を読んでこれを理解するのは難しいと思いますが、簡単には証明書ポリシーが最上位の中間 CA 証明書から繋がっているものだけがポリシー処理の結果有効な証明書チェーンになります。

証明書ポリシーのためのワイルドカード(anyPolicy)が使えたり、指定した回数処理をスキップしたり(policyConstraints, inhibitAnyPolicy)、あるポリシーを別のポリシーに置き換えたり(policyMappings)など様々な複雑なポリシー処理を加えることができますが、簡単には「証明書ポリシーが繋がっていること」が求められます。

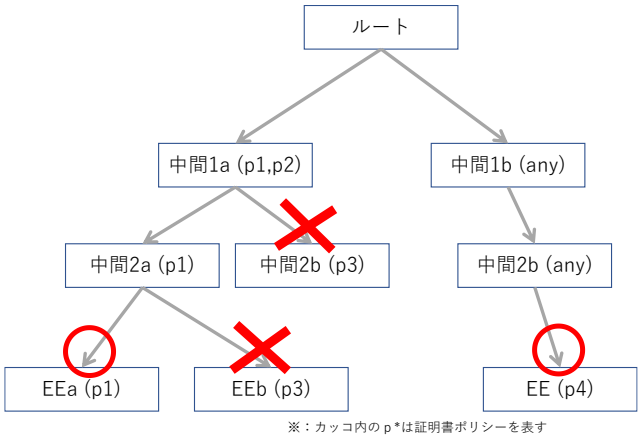

図 A.6-1 ポリシーが繋がる証明書チェーンの選択

そして、結果として残った有効な証明書ポリシーの中から、受け入れ可能な証明書ポリシーのもののみを選び、それが有効なチェーンになります。

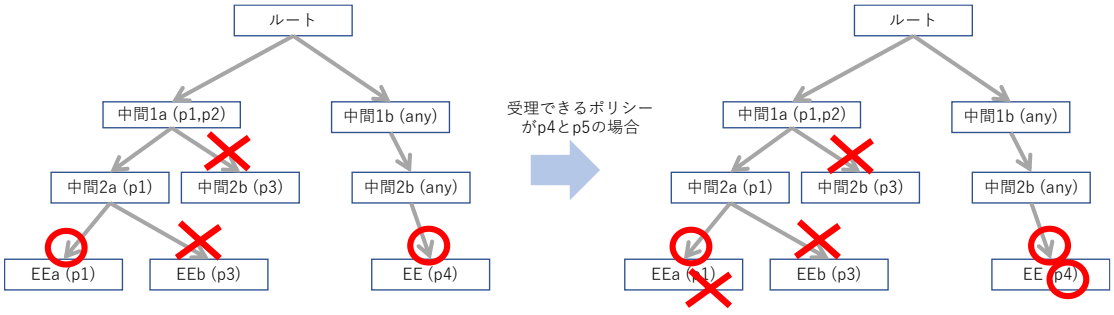

図 A.6-2 繋がった証明書チェーンからポリシーが受容可能なものを選択

ポリシー処理を正しく実装しない、もしくはポリシー処理の一部、もしくは全てをサポートしないソフトウェアもあるので、ポリシー処理を必要とするアプリケーション、またはドメインでは自分が使用するソフトウェアのポリシー処理のサポート状況を確認して検証する必要があります。

もし余裕があれば、NIST で開発された PKI テストスイート(PKITS)[16] を用いて、ソフトウェアの証明書検証におけるポリシー処理対応状況を確認し、自分が適用しようとしている要件にあっているか確認するとよいでしょう。

(2) 共通の証明書ポリシーOID の処理

以前はブリッジ CA を使用している場合など、限定された領域でしか厳密なポリシー処理を必要としないと考えていましたが、この数年、正しいポリシー処理を必要とするケースが増えてきたように思います。例えば以下のようなケースです。

① EU eIDAS の署名用証明書、e シール用証明書、QSCD 利用の有無など、適格かそうでないかのレベル等、証明書ポリシーによりこれらを識別が必要なケースが出てきた。

② CABF の S/MIME 用証明書に関する基本要求(SMIMEBR)[17] においても 12 種類のプロファイルを区別しており、これを識別しなければならないケースがある。

③ 日本の e シール用証明書、署名用証明書についても保証レベルに応じ共通の証明書ポリシーOID を割り当てることが検討されており、これが導入される可能性があるかもしれない。

④ EV SSL 証明書、コード署名証明書なども特別な証明書ポリシーOID を使うことで識別している。

前項のポリシー処理が正しく行える検証ソフトウェアであれば、これらを正しく処理することができます。

A.6.4 業界やアプリケーションに依存するその他の拡張等の追加検証

対象とする領域によってはさらに追加で証明書の検証をしなければならないケースがあります。

① EU 適格証明書で qcStatements 拡張がある場合に、検証しようとしているものが署名か e シールか、QSCD を必要とするか、金融機関の PSD2 指令に基づく属性等、検証が必要なケースがあります。

② 医療従事者の資格等が subjectDirectoryAttributes 拡張に記載されており、これの検証が必要なケースがあります。

③ 主体者識別名に組織の一意な識別番号である organizationIdentifier があり、公開されている

[16] NIST Public Key Infrastructure Testing, X.509 Path Validation Test Suite
https://csrc.nist.gov/Projects/pki-testing/x-509-path-validation-test-suite

[17] CA/Browser Forum, Baseline Requirements for The Issuance And Management Of Publicly-Trusted S/MIME Certificates Version 1.0.1

ベースレジストリ等で確認ができる場合に、これを検証する必要があるケースがあります。

A.6.5 Short Lived 証明書

リモート署名サービス等で、利用者が年に 1 度とか数ヶ月に 1 回とか非常に少ない頻度で署名をする場合に、利用者の署名鍵をずっと預かっておくのは HSM のリソース的に負担になることがあります。署名が必要な時に非常に短い期間、例えば数時間等の有効期間の短い証明書を発行し署名するというケースがあり、これは Short Lived 証明書と呼ばれています。Short Lived 証明書の数時間といった短い有効期間を超えて検証しなければならない場合には、署名タイムスタンプを利用することにより検証が可能になります。

Short Lived 証明書は有効期間が短いために最初から証明書失効を行わない前提であるケースがあり、その場合に CRL 配布点拡張、AIA OCSP 拡張がないケースがあります。そのようなケースでは、Short Lived 証明書であることを確認して失効検証をスキップできる機能が必要となります。

Short Lived 証明書の利用については Cloud Signature Consortium(CSC)[18] においても標準化が検討されています。

A.6.6 失効検証の注意点

証明書の検証を行う際に、利用している証明書検証のソフトウェアがどのような失効情報のモデルに対応しているのか確認してから利用しなければ、期待した検証結果が得られないケースがあります。

(1) CRL の発行モデル

CRL の発行方法には幾つかのモデルがあります。

① CRL を、証明書を発行する CA の鍵で署名し CRL が 1 つしかない最も一般的なモデル

② 発行者の識別名は同じだが、CRL を署名する鍵と証明書を発行する鍵を分けているケースで、CRL を発行した証明書を辿ってトラストアンカーに含まれていれば有効とするケース

③ CRL/ARL モデル:エンドエンティティ検証用の CRL と CA 証明書検証用の CRL(=ARL)を分けているケース。公的個人認証証明書、GPKI の証明書などがこれに該当するケースがあります。

④ 分割 CRL モデル:大量のユーザーを抱える CA が CRL を発行する場合に、CRL の肥大化を防ぐため、数千人等対象ユーザーを区切った CRL を分割して発行するケース。数万人といった従業員を抱える場合に適用されることがあります。

⑤ Delta CRL モデル:失効検証を極力リアルタイムでできるようにするために、一般的な周期で全体の CRL(=フル CRL)を発行すると共に、短周期でその差分となる CRL(=Delta CRL)を発行するケース。Microsoft の CA 製品で発行することが可能です。

⑥ OCSP のみのモデル:一つの CA が大量の証明書を発行し、また失効の件数も多い場合に

[18] Cloud Signature Consortium https://cloudsignatureconsortium.org

CRL を発行せず、失効情報として OCSP レスポンスしか提供しないケースがあります。Let's Encrypt などがこの方式を採用しています。

使用する証明書検証器が必要とする CRL モデルに対応しているか確認する必要があります。

(2) OCSP の発行モデル

OCSP レスポンス(RFC 6960)[19] の発行には 2 つのモデルがあります。

① 直接(direct)モデル:証明書を発行する CA の秘密鍵で OCSP レスポンスも署名し発行するケース。その場合、CA 証明書の keyUsage には digitalSignature ビットを含む必要があります。これまで CA の秘密鍵を必要以上に頻繁に使わない方がセキュリティ上、良いとして移譲モデルを採用する CA が多く、ユーザーの少ない小さい PKI でしか見ることが無かったですが、近年 Let's Encrypt など直接モデルの OCSP を採用する CA があるようです。

② 移譲(delegated)モデル:CA の秘密鍵で OCSP レスポンスを署名/発行するのではなく、OCSP レスポンダーと呼ばれる OCSP レスポンス発行専門の発行局に OCSP レスポンスの発行を移譲するケース。多くの OCSP に対応するパブリック CA がこのモデルを採用しています。

使用する証明書検証器が必要とする OCSP 発行モデルに対応しているか確認する必要があります。

(3) 失効情報の掲載期間

CRL や OCSP で失効情報を取得する際に、認証局によって失効情報を提供する期間が異なり、また、その情報は CP/CPS にも掲載していないことが多いことから、証明書検証を使ったシステムではこれに配慮する必要があります。特に、長期署名を扱うケースでは失効情報の掲載期間に配慮した設計を行う必要があるでしょう。

CRL の失効情報の掲載期間については下記の 2 つのモデルがあります。

- CRL の失効情報を当該証明書の有効期間のみしか掲載しないモデル:
  CRL の肥大化を防ぐため、証明書の有効期間しか失効情報を掲載しないケースがあります。多くの CA がこのモデルを採用しています。
- CRL の失効情報を当該証明書の有効期間を超えて掲載するモデル:
  CRL に掲載する失効情報を当該証明書の有効期間を超えて提供する CA があり、これが CA 証明書の有効期間、永続的に提供される場合と期間限定で提供するケースがあります。一部の認定認証事業者の CA では永続的に失効情報を掲載しています。

[19] RFC 6960 X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP

OCSP についても同様に失効情報の提供期間に違いがあります。

- 検証対象の証明書の有効期間を超えても失効の有無を確認できる有効な OCSP レスポンスを返す CA
- 検証対象の証明書の有効期間を超えると OCSP レスポンスを返さなくなる CA

(4) 長期署名における OCSP のメリットと注意点

本書「3.2.3(2)失効確認の注意点」や「附録 A.8 猶予期間の運用」でも述べられていますが、長期署名に格納し検証時に使用する CRL の取得タイミングは非常に悩ましい問題であり、長期署名を用いたシステムにおける、検証情報の追加、署名延長などの運用を難しいものにしています。

検証時刻に、証明書チェーンの証明書を CRL 失効検証するのに、もし失効された可能性があったとしても十分な失効反映期間を待った上で、長期署名の検証情報として格納し、検証する側もこれに配慮しなければならないというのが猶予期間の考え方です。CRL の猶予期間で特に問題なのは以下です。

- CRL 猶予期間については、署名用証明書だけでなく、CA 証明書やタイムスタンプの失効検証用の CRL についても本来考える必要がありますが、CA 証明書やタイムスタンプの検証用の CRL 発行周期は数週間や 1 ヶ月など非常に長いケースが多く、また、失効時には CRL 定期発行外のものを使用しなければならないケースもあります。
- CRL の発行頻度は CA や認証ベンダーによっても異なり、また、CP/CPS にも発行周期が記載されていなかったり、必ずしも CRL の thisUpdate、nextUpdate から判断つかず記載よりも短い頻度で発行されるケースもり、酷い場合には、運用過程で CRL の発行周期が気づかずに変更になっているケースもあります。これらの問題が長期署名の検証情報の追加やアーカイブの観点で運用を難しくしています。

以上のような猶予期間の問題は OCSP レスポンス(RFC6960)を使うことで解決できるのではないかと考えています。

長期署名の失効検証情報として OCSP レスポンスを使うことのメリットは以下の通りです。

- CRL はいつ取得したかという情報は CRL 自体には記載されていません、OCSP レスポンスには取得した時刻が記載されているため、十分な猶予期間が取れていたのかという証明もしやすく、また取得時点で失効していなかった証明もしやすいため、長期署名への検証情報の付加、アーカイビングのタイミングなどのシステム設計、運用設計がしやすくなります。これは特に、CRL 発行周期の長い上位の CA 証明書の失効検証でメリットとなります。
- 発行証明書数やユーザー数が多い CA の場合には、CRL のデータサイズは大きなものになってしまい、長期署名への失効検証情報の付加に躊躇するが、OCSP レスポンスの場合には

サイズがほぼ同じ大きさで CRL に比べて小さいケースが多く、ストレージの容量も削減でき、システムのリソース計画を立てやすい。

数万、数十万といった従業員を抱えるエンタープライズ向けの CA の場合、従業員の定期配置転換などで大量の失効が発生する場合、公的個人証明書などの全市民、国民向けの証明書でそもそものユーザー数が多い場合などは、長期署名で使用されることを見越して上位の CA 証明書も含め OCSP の提供を合わせて CA 設計されることが望まれます。

メリットの一方で、OCSP を利用する際にはいくつかの注意点もあります。

- OCSP が移譲(delegated)モデル、OCSP レスポンダーを使うモデルで運用されている場合、OCSP レスポンダー証明書およびそれらの証明書チェーンや検証情報も含めて検討しておかなければなりません。
- CA によりますが、失効検証対象の証明書が期限切れになった後、OCSP レスポンスを返さなくなるケースがあります。前述も考慮し OCSP レスポンスはなるべく早めに取得し格納しておくのが良いと考えます。
- OCSP レスポンダー証明書に ocspNoCheck の拡張が記載されていることがありますが、これを検証しなくて良いと勘違いしている人、実装がまれにあります。ocspNoCheck は「OCSP レスポンダー証明書の失効検証はしなくてよい」つまり、これ以外の認証パス検証、上位の CA 証明書の検証や失効検証も含め必要となるので注意してください。

OCSP はエンドエンティティ証明書にしか提供されないケースもありますが、長期署名のシステムの運用を考えると証明書チェーンの全てで OCSP が提供されていることが望ましく、CA を選択可能なケースではそのような CA を選ぶと良いでしょう。

(5) 失効情報の提供期間に関連した拡張

本項では、長期署名に影響ある失効情報の提供期間に関連した拡張について紹介します。

① expiredCertsOnCRL CRL 拡張(oid=2.5.29.59)
- ITU-T X.509 の 9.5.2.8 節で規定されている CRL 拡張（RFC 5280 には記述なし）
- 値に日付設定できる。
- この拡張に記載された日時以降に期限切れになった証明書の失効情報を CRL に含むことを示す。
- 証明書の期限切れ後は(記載された)失効状態は変更してはならない。

② archiveCutOff OCSP singleExtensions 拡張(oid=1.3.6.1.5.5.7.48.1.6)
- RFC 6960 OCSP の 4.4.4 節(のみ)で規定されている OCSP singleResponse 拡張
- 値として GeneralizedTime 時刻を持つ

- 証明書の期限切れがその時刻以降なら OCSP 失効状態を出せる時刻
- 例えば「証明書期限切れ後、7 年間は失効状態を OCSP で返す」ポリシーであったとすると、OCSP レスポンスの producedAt フィールドの値(=事前生成でなければ現在時刻)の 7 年前の日時が archiveCutOff 拡張に記載される。
- 証明書が現在、失効しているか、期限切れかどうかは archiveCutOff の値には関係ない。

③ expiredCertsRevocationInfo TrustedList 拡張

- ETSI TS 119 612 TrustedList[20] の 5.5.9.1 節で規定された TrustedList 拡張
- 失効状態をいつまで提供するのかを示す。

これらの拡張は ETSI EN 319 411-2「証明書を発行する TSP のポリシーとセキュリティ要件 Part2 EU 適格証明書を発行する TSP の要件」[21] でも参照されています。

[20] ETSI TS 119 612 V2.1.1 (2015-07) Electronic Signatures and Infrastructures (ESI); Trusted Lists

[21] ETSI EN 319 411-2 V2.3.1 (2021-05) Electronic Signatures and Infrastructures (ESI); Policy and security requirements for Trust Service Providers issuing certificates; Part 2: Requirements for trust service providers issuing EU qualified certificates

A.7 タイムスタンプのトラスト

A.7.1 タイムスタンプ（RFC 3161）

電子データの存在証明をするタイムスタンプは、PKI の技術を利用しています。
時刻認証局（タイムスタンプ局 TSA）が管理する署名鍵で、デジタル署名される対象は、TSTInfo と公開鍵証明書（Certificate）です。TSTinfo には、対象情報のハッシュ値、署名時刻、時刻精度、発行ポリシー、TSA 情報等が含まれており、対象情報が、署名時刻に存在していたことを発行基準情報と共に確認できます。公開鍵証明書は、署名に使用した署名鍵の証明書が含まれ、確かに TSA が管理している署名鍵でデジタル署名したことを検証できます。

RFC3161 のプロトコルとデータ構造を図に示します。

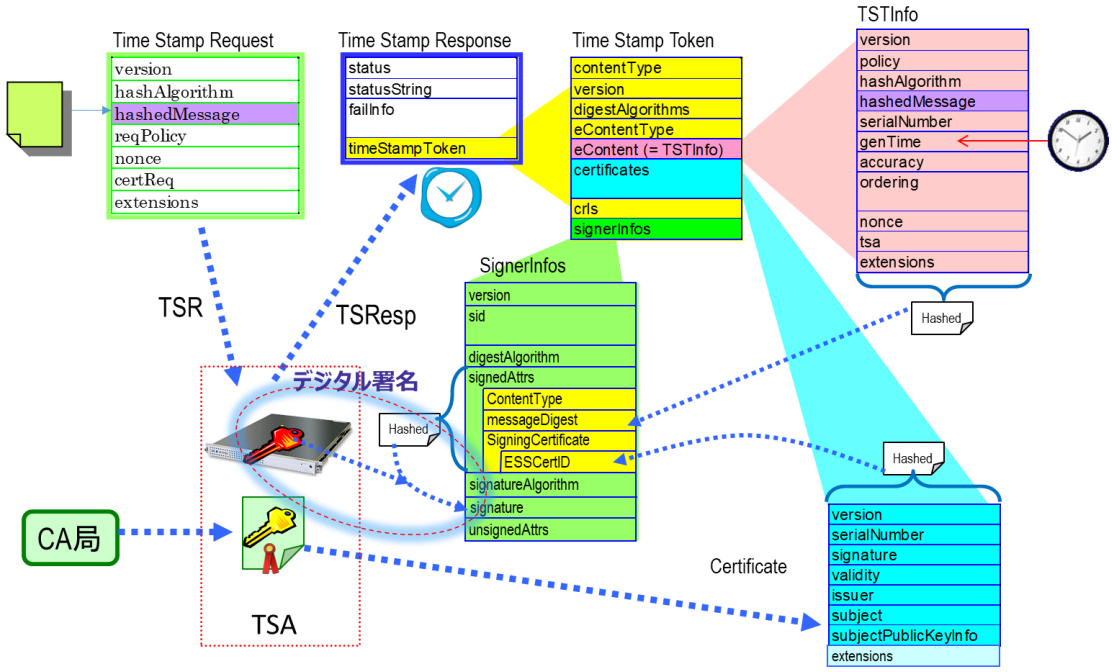

図 A.7-1　RFC3161 タイムスタンプのプロトコルとデータ構造

A.7.2 トラストを確保する重要な要素

タイムスタンプの信頼性を保証するためには、
・信頼のおける時刻であること
・署名鍵が正しく管理されていること
を将来にわたって証明できる必要があります。

（1）信頼のおける時刻

RFC3161 では、タイムスタンプとして付与する時刻についての規定はありません。時刻は、

国際的に共通化されている情報で、事象の尺度としてあたりまえに利用されているラベルです。そしていつでもどこでも、あらゆる時計から取得が可能です。このことは、タイムスタンプ局が、任意に設定が可能であることを意味します。タイムスタンプは存在証明をするエビデンスなので、その時刻は将来にわたって正しかったことを証明できる必要があります。

時刻認証局は、信頼のおける時刻源から時刻情報を入手し、まちがった時刻が利用されていないことを管理し、将来にわたって保証する必要があります。

(2) 署名鍵の管理

タイムスタンプは、デジタルデータの存在時刻の証明目的でさまざまな用途で利用されることから、付与する利用者に限らず、付与されたタイムスタンプを検証し、信用して利用する依拠者も含め、不特定多数の影響者が存在します。

署名鍵が漏洩、紛失した場合、その署名鍵を利用して発行されたタイムスタンプはすべて信用できなくなるため、社会的に大きな問題を引き起こす可能性があります。

このため、署名鍵の管理は厳重に行われる必要があります。タイムスタンプの署名鍵は、事業者として管理することから、認証局は個人への発行に比べて、より長期な有効期限の電子証明書を発行できます。

A.7.3 総務大臣認定

上記の(1)(2)を TSA 自らが証明することは困難です。

そのため、日本においては、総務大臣による認定制度が運用されていて、タイムスタンプを発行する事業者の技術および運用について、指定調査機関により適合性の監査をすることで保証されています[22]。

総務大臣認定では、時刻認証業務の認定に関する実施要項[23]にて以下のように規定されています。

(1) 信頼のおける時刻

第 6 条にて、時刻源を日本標準時の報時業務が定められている国立研究開発法人情報通信研究機構[24]の UTC(NICT)に対して±1秒以内で同期することが定められています。

また、第 7 条 1 項で、その時刻の品質を証明しうるため以下の記録について完全性と機密性を保ちながらタイムスタンプが有効である期間、保管することを定められています。

① TSA の時計が UTC(NICT)に同期されるまでに経由する各機器における時計間の時刻差を測定した時刻同期ログ

② 鍵ペアの生成・失効記録及び秘密鍵廃棄の記録

③ 認定業務に関わるシステムの動作異常の記録

[22] 時刻認証業務の認定に関する規程(令和 3 年総務省告示第 146 号)

[23] https://www.soumu.go.jp/main_content/000743330.pdf

[24] 国立研究開発法人情報通信研究機構法第 14 条 1 項三号

④ その他時刻の品質を管理又は証明するために必要な記録

さらに、同条 3 項にて、この精度を満たしていないタイムスタンプを発行しないシステムであることも定められています。

(2) 署名鍵の管理

第 10 条にて、HSM(ハードウェアセキュリティモジュール)に保護することが規定されています。

そして、安全に管理するため具体的な規定が、第 12 条で定められています。

① 複数人管理のもと信頼できる鍵生成装置によって生成し HSM 内に保管
② 秘密鍵(署名鍵)のバックアップ禁止
③ 複数人の権限を有する者が揃わない限りは HSM の持ち出し等禁止
④ HSM 内部でデジタル署名
⑤ 有効期間終了、失効又は危殆化した場合等は、当該秘密鍵を廃棄
⑥ CRYPTREC 暗号リスト等の最新の安全性評価を基に、有効期間及び活性化期間をあらかじめ適切に定め定期的に更新

総務大臣による認定制度にて認定されている事業者の TSA 公開鍵証明書は、総務省の Web[25] にて公開されており、認定された事業者から発行されたタイムスタンプであることを確認できます。

また、認定タイムスタンプを利用したサービスは、(一財)日本データ通信協会にて「認定タイムスタンプを利用する事業者に関する登録制度[26]」で登録され、認定タイムスタンプ利用登録マークを確認することで識別ができます。

[25] https://www.soumu.go.jp/main_sosiki/joho_tsusin/top/ninshou-law/timestamp.html
[26] https://www.dekyo.or.jp/touroku/

A.8 猶予期間の運用

本文の 3.2.3(2)失効確認の注意点で、署名やタイムスタンプの検証において、猶予期間の考慮が必要と述べましたが、その実運用上の課題と留意事項について説明します。

A.8.1 猶予期間の課題

署名の検証において、署名に利用された証明書(署名鍵)は有効期間内でも署名鍵の紛失、盗難、漏洩によって失効することがあることから、署名時の証明書の状態が争点となることが想定されます。

証明書の状態として以下の 3 つの状態が考えられます。

①無効(有効期限切れ・失効情報に失効が記録されている)

②不明(情報不足)

③有効(暗号アルゴリズムの危殆化および署名鍵の漏洩が無く証明書が有効期間内)

ここでは、有効とも無効とも判断ができない「②不明」状態について考察します。

証明書の状態を確認する方法として、証明書の有効期限を確認することに加え、認証局から定期的に発行・公開される失効情報を取得して確認する方法がありますが、まず、失効情報が取得されていない状態は、当該証明書について「不明」状態であると言えます。

次に、失効情報が取得された状態にあって、失効情報に最新の失効記録が反映されていない状態も「不明」状態であると考えることができます。

この署名時の証明書の状態が「不明」であることのリスクとは、署名時に失効(鍵の漏洩発生もしくは失効申請中)していたにもかかわらず、失効情報への反映が間に合わず、署名が行われたことが事後に発覚し、署名の効力について争われる可能性があることです。これは、長期署名データに当該証明書の有効性を示す情報として最新の失効記録が反映されていない失効情報が格納されてしまうことにより発生し得る潜在的リスクと言えます。

上記のリスクを軽減する対策方法の1つとして、署名後に一定期間(以下、猶予期間:Grace Period)経過した後に、新たに発行された失効情報を取得して長期署名データに埋め込むという方法が考えられました。

つまり、猶予期間経過した後に新たに発行された失効情報ならば、署名時に仮に失効申請していたとしても、その失効が失効情報に反映されているだろう、したがって、失効が失効情報に反映されているタイミングに取得した失効情報に当該証明書の失効が記録されていないということは、そのタイミングにおいての署名(証明書)は「③有効」状態であっただろうと推測され得るという考え方に基づいています。

さて、この「猶予期間」とは、署名鍵の所有者が自己の署名鍵の漏洩に気づき、認証局に失効申請してから、その失効が失効情報に反映され、それを取得するまでの期間と考えることができま

す。

一般的に、認証局に失効申請してから失効情報に反映されるまでの期間として、失効申請タイミングが認証局の営業時間内であれば、早くて当日中、もしくは翌営業日中と想定されますが、一律に何日間や何時間のように決まった答えが無い上に、署名鍵の所有者が自己の署名鍵の漏洩に気づくまでの期間は定義できないことがこの方法の課題です。

したがって、この猶予期間が長ければ長いほど、署名時において失効していなかった、つまり、署名（証明書）は有効であったということの「確からしさが増す（不明の度合いが下がる）」ということしか言えません。

また、猶予期間を長く取りすぎて署名に利用された署名者証明書の有効期限後に発行された失効情報では意味をなさないので注意が必要です。特に長期署名データを生成する場合は、署名者証明書の有効期限切れの数日前（猶予期間分）までしか利用しないことが推奨されます。

一方、長期署名データの生成において、猶予期間を経ないと最新の失効情報を格納した長期署名データが完成しないということももう一つ課題と言えます。ただし、この点については、認証局の運用にも依存しますが、失効情報の形態として、一定周期で更新される CRL（Certificate Revocation List：証明書失効リスト）に比べ、失効状態が即時に反映される OCSP（Online Certificate Status Protocol）レスポンスを失効情報として長期署名データに格納することで、長期署名データの完成までの期間を短縮することができる場合があります。

A.8.2 猶予期間の運用上の留意事項

リスクは発生可能性と影響度で評価されますが、上述したリスクおよびそのリスク軽減策としての猶予期間を設ける方法の課題を考慮した上で、総合的なリスク評価の結果、リスクを受容する、つまり、猶予期間を設けないという選択肢も考えることができます。

例えば、電子契約のようなアプリケーションサービスの分野では、署名対象データに対して、複数人が署名を行う場合があり、最初の署名者が署名を行ってからその署名の失効情報の取得のために猶予期間待ってから、次の署名者が署名を行ってという仕組みは、いつでも・どこでもできるというデジタルで行うメリットおよびユーザー体験を大きく棄損することから、猶予期間を設けない場合のリスクを受容するという判断が行われることがあります。もちろん、その場合でも、署名鍵の管理が適切に行われるシステムやサービスを使うなどの手段によりリスクの発生可能性の低減措置を行う、発生した場合の金額等の影響度が許容できるなどの判断が行われていることが前提となります。

最終的には、署名データの作成者および検証者が双方合意できる方法で、適切な失効情報を格納することが望まれますが、少なくとも、利用予定の署名付与アプリケーションや署名検証アプリケーションが、猶予期間を考慮することが可能なのかを確認することが必要です。

尚、署名者証明書の上位の CA 証明書やタイムスタンプ用の証明書についても同様に猶予期間に配慮する必要があります。CA 証明書のための CRL の発行周期が数ヶ月と長いような場合には、長期署名フォーマットに猶予期間に配慮した証明書検証情報の格納が困難なケースもあり、予め署名者、検証者の間で明確にしておく必要があるでしょう。

A.8.3 猶予期間と認証局

GPKI や LGPKI のようにルート証明書の鍵更新に際して、新旧のルート証明書をリンク証明書で関係をつなぎ、失効情報が新ルート証明書で発行されても旧ルート証明書から発行された署名者証明書の検証が行える仕組みが運用されていますが、長期署名データを生成および検証しているアプリケーションにおいて猶予期間をチェックしている場合に次の問題が発生する場合があります。

通常、長期署名データに格納される署名者証明書やルート証明書、失効情報などは、(署名タイムスタンプ時刻に基づく)署名時刻において、証明書の有効期間および失効状態の確認などの観点で「有効」である必要があります。

ところが、旧ルート証明書から発行された署名者証明書で署名を行い署名タイムスタンプを付与した状態の長期署名データ(XAdES-T レベル)が、猶予期間を考慮して新しく発行される失効情報を待っているタイミングにおいて、新ルート証明書の生成とその新ルート証明書による失効情報の発行がほぼ同時に行われた場合、新ルート証明書の有効期間の発行時刻(notBefore)が署名タイムスタンプ時刻の後、つまり、署名タイムスタンプ時刻において有効期間外の(新)ルート証明書と判断されてしまうという問題が発生することになります。

もし、長期署名データの生成における猶予期間の考え方を認証局が認識して、新ルート証明書の生成が行われた後、十分な日数を経た後に、新ルート証明書から失効情報を発行する運用を開始するということが行われれば、上記のような問題は発生しなかったと考えられます。

このように、署名者や検証者だけでなく、認証局においても、長期署名データの生成や検証に関する理解、および運用的な協力(事前の告知や新ルート証明書の公開など)があることが望ましいと考えられます。

作成メンバー（五十音順）

漆嶌 賢二（GMO グローバルサイン株式会社）
小川 博久（株式会社三菱総合研究所）
柴田 孝一（セイコーソリューションズ株式会社）
西山 晃（フューチャー・トラスト・ラボ）
政本 廣志（日本ネットワークセキュリティ協会 電子署名 WG）
宮崎 一哉（三菱電機株式会社）
宮地 直人（有限会社ラング・エッジ）
村尾 進一（有限会社ラング・エッジ）