ISBN978-4-89078-655-8 C2055

電子文書長期保存ハンドブック

平成19年3月

次世代電子商取引推進協議会

電子文書長期保存ハンドブック

平成１９年３月

次世代電子商取引推進協議会

序文

2005 年 4 月に e 文書法が施行され、これまで民間企業において紙での保存が義務付けられていた税務関連等の文書（書類や帳票等）をデジタルデータで保存することが可能となった。長期保存を考えると、電子署名を行う人（署名者）とそれを数十年後に検証する人（検証者）が同じシステムを利用するとは限らず、非標準のシステムの導入はひとつのベンダに制約されるだけでなく、そのベンダのサービス停止によって、保管していた文書データの利用が困難になる可能性がある。

ECOM では 2000 年度から文書や電子署名文書の保存技術に関するガイドラインの作成や各種調査研究を行ってきた。これらの活動を通じて幾つかのベンダが実際の製品を開発したり、特定のプロジェクトでこれらの技術を用いるようになってきた。

そこで前年度はこれまでの成果を整理し、RFC3126 や ETSI の署名フォーマットのプロファイルの規定を策定し、このフォーマット文書の保存管理要件やこのプロファイルに基づいた相互運用性テスト要件をまとめ、これらのテスト要件に基づく製品の相互運用性テストを実施した。

今年度は、この成果に基づいて２つの JIS 原案*を作成し、METI に対して JIS 提案を行った。それとともに、電子署名の標準化をいっそう進めるためには理解しやすいハンドブックが必要であり、電子署名ハンドブックの作成を行った。さらに、日本版 SOX 法で議論されているように署名付文書の保存について総合的に対応する必要があり、見読性の問題、文書フォーマットの問題について検討を加えた。

本報告書は、かなり深く検討を行っていることもあり、読者の理解を考え次のとおり 3 部構成にしている。

第一部　デジタル署名ハンドブック

第二部　電子文書長期保存

第三部　長期署名プロファイルの JIS 化、相互運用実験、及び関連する国際調査

今年度の活動は、会員各位の多大なご協力のもとに、長期保存フォーマット実装製品の相互運用性確保のために大いに寄与することができた。本報告が広く国内外から参照され、長期保存フォーマットの普及促進の一助になれば幸いである。

平成１９年３月

次世代電子商取引推進協議会

---

* 「暗号メッセージ構文を利用した電子署名(CAdES)の長期署名プロファイルに関する要求事項」
「拡張可能なマーク付け言語を利用した電子署名(XAdES)の長期署名プロファイルに関する要求事項」

目　次

第一部

デジタル署名ハンドブック

まえがき

本ハンドブックは、標準形式の署名付きデータの生成及び検証に関わる処理系を正確に実装するための参考書である。対象読者としては、同処理系を開発する技術者を想定している。

ECOM においては、2000 年度より長期署名に関する調査･研究を実施してきた([42]-[54])。2007 年中にはそのプロファイルの JIS 化が見込まれているが、この規格書を参照しただけでは処理系の実装は難しい。規格書が参照する規格書に立ち返って処理系を実装することは不可能ではないであろうが、それらを誤解なく読み解くにはかなりの労力を要する。本ハンドブックは、関連する各種規格書の内容を総括し、解説を加えることにより、長期署名、あるいは長期署名に拡張可能な署名付きデータを正確に取扱う処理系の実装を助けるものである。

本ハンドブックの記述範囲は次のとおりである。

(1) CMS SignedData 及び、その拡張フォーマットである CAdES（CMS Advanced Electronic Signatures）の構造。CAdES については、RFC 3126[4]で示されるフォーマットと ETSI TS 101 733 V1.7.3 (2007-0) [13]で示されるフォーマットの両者を対象とする。
(2) XML 署名及び、その拡張フォーマットである XAdES（XML Advanced Electronic Signatures）の構造。XAdES については、ETSI TS 101 903 V1.2.2 (2004-04)[20] で示されるフォーマットと ETSI TS 101 903 V1.3.2 (2006-03)[22] で示されるフォーマットの両者を対象とする。
(3) CAdES と XAdES の生成手順
(4) CAdES と XAdES の検証手順

なお、公開鍵証明書、失効情報、タイムスタンプトークン、署名ポリシの検証処理の詳細については参考文献を参照願いたい。

本ハンドブックが、長期署名を含むデジタル署名の相互運用性確保に、延いてはそれらの普及に資することができれば幸いである。

第一部　デジタル署名ハンドブック

I. デジタル署名フォーマット詳細

1. CAdES

長期署名フォーマットには CMS SignedData に基づく形式と、XML 署名に基づく形式がある。本節では CMS に基づいた CAdES について説明する。

1.1. CAdES(CAdES-BES, CAdES-EPES)

本節で説明する狭義の CAdES は、署名対象文書に署名者の署名情報を付与したものであり、CMS や PKCS #7 の署名データ(SignedData)の基本形である。

1.1.1. 基本構造

電子署名の添付された文書(電子署名付文書)の基本構造を図 1.1.1.1 に示す。

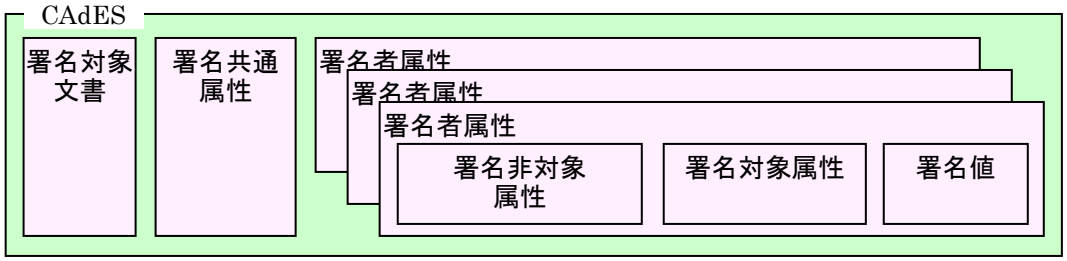


図 1.1.1.1: 電子署名付文書の基本構造

電子署名付文書は、署名対象の文書に対し、署名に関わる共通の属性と署名値を含む 1 つあるいは複数の署名者属性で構成される。署名対象文書が、電子署名付文書の基本構造内に含まれている場合と、分離されている場合(図 1.1.1.2)がある。

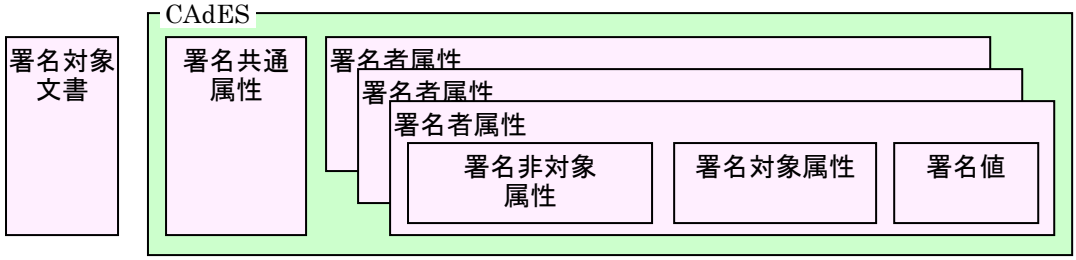


図 1.1.1.2: 署名対象文書が電子署名付文書の基本構造から分離されている場合

また、署名対象属性に署名ポリシを含まない場合(CAdES-BES：Basic Electronic Signature)と含む場合(CAdES-EPES：Explicit Policy Electronic Signature、図 1.1.1.3)とを区別することがある。

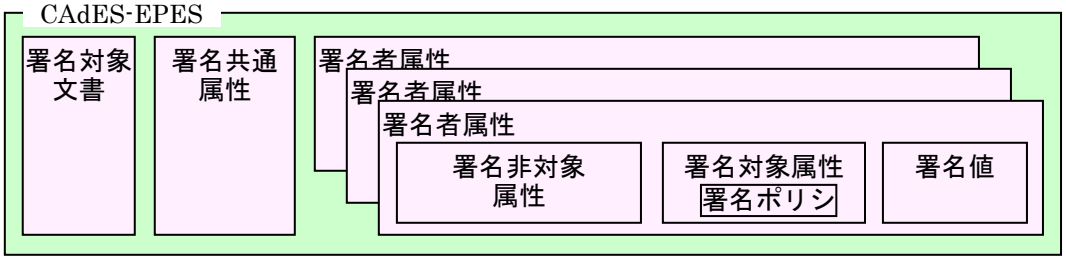


図 1.1.1.3: 署名ポリシを含む電子署名付文書の構造

以降、CAdES(CAdES-BES、CAdES-EPES の両者を含む)の詳細構造について説明する。

1.1.2. CAdES のトップレベルの構造

CAdES のトップレベルの構造を次に示す。

```
RFC 3852[5] 3 節 General Syntax より
ContentInfo ::= SEQUENCE {
            contentType  ContentType,
            content      [0] EXPLICIT ANY DEFINED BY contentType }
```

ただし、contentType の型は

```
RFC 3852[5] 3 節 General Syntax より
ContentType ::= OBJECT IDENTIFIER
```

であり、CAdES での値は次のとおりである。

```
RFC 3852[5] 5.1 節 SignedData Type より
id-signedData OBJECT IDENTIFIER ::= { iso(1) member-body(2)
                                    us(840) rsadsi(113549) pkcs(1) pkcs7(7) 2 }
```

content には次に示す SingedData 型の値が格納される。

1.1.3. SingedData

SingedData の型定義は次のとおりである。

```
RFC 3852[5] 5.1 節 SignedData Type より
SignedData ::= SEQUENCE {
  version          CMSVersion,
  digestAlgorithms DigestAlgorithmIdentifiers,
  encapContentInfo EncapsulatedContentInfo,
  certificates     [0] IMPLICIT CertificateSet OPTIONAL,
  crls             [1] IMPLICIT CertificateRevocationLists OPTIONAL,
  signerInfos      SignerInfos }
DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier
SignerInfos ::= SET OF SignerInfo
```

各フィールドの型及び値は次のとおりである。

1.1.3.1. version フィールド(CMSVersion 型): SignedData の版番号

本フィールドは CMS のバージョン番号を保持する。CAdES では一般的には 1 か 3 となる。CMSVersion の型定義は次のとおりである。

```
RFC 3852[5] 12.1 節 CMS ASN.1 Module P.47 より
CMSVersion ::= INTEGER  { v0(0), v1(1), v2(2), v3(3), v4(4), v5(5) }
```

値は次の規則によって決定する(MUST[10]5.1)。

```
IF ((certificates 要素が存在) AND
   (other 型の certificate が存在)) OR
   ((crls 要素が存在) AND
   (other 型の crls が存在))
THEN version MUST be 5
ELSE
   IF (certificates 要素が存在) AND
      (バージョン 2 の属性証明書が存在)
   THEN version MUST be 4
   ELSE
      IF ((certificates 要素が存在) AND
         (バージョン 1 の属性証明書が存在)) OR
```

```
        (SignerInfo 型のバージョンが 3。つまり SignerInfo の SignerIdentifier
          が subjectKeyIdentifier の場合。) OR
        (encapContentInfo の eContentType が id-data 以外)
    THEN version MUST be 3
    ELSE version MUST be 1
```

1.1.3.2. digestAlgorithms フィールド(DigestAlgorithmIdentifiers 型)

本フィールドは、CMS SignedData で使用される全てのハッシュアルゴリズムの識別子を保持する。署名検証をワンパスで(署名データを先頭から 1 回走査するだけで)行いやすくするためのフィールドである。

一人、もしくは複数人の署名者によって使用される全てのハッシュアルゴリズムをこのフィールドに格納しておくことが想定されているが、規定はされておらず、本フィールドに格納するハッシュアルゴリズムの数は任意である(0 でもよい)。

本フィールドは署名検証を行う機器のメモリが非常に少ないという想定の下で設けられたもので、現在のコンピューティング環境では存在意義が薄くなっており、後方互換のために残っているフィールドだと言える。

署名時に使用されたハッシュアルゴリズムを署名検証時に取得する場合、現在は後述する SignerInfo 内の digestAlgorithm フィールドを直接参照するのが一般的である。

DigestAlgorithmIdentifiers の型定義は次のとおりである。

```
RFC 3852[5] 5.1 節 SignedData Type より
DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier

RFC 3852[5] 10.1.1 節 DigestAlgorithmIdentifier より
DigestAlgorithmIdentifier ::= AlgorithmIdentifier

RFC 3280[6] 4.1.1.2 節 signatureAlgorithm より
AlgorithmIdentifier  ::=  SEQUENCE  {
     algorithm               OBJECT IDENTIFIER,
     parameters              ANY DEFINED BY algorithm OPTIONAL  }
```

1.1.3.3. encapContentInfo フィールド(EncapsulatedContentInfo 型)：署名対象文書

encapContentInfo は、CMS SignedData に署名対象文書(データ)を署名フォーマット内に含める場合、カプセル化して格納するフィールドである。含めない場合には、データ型のみが含まれる。

encapContentInfo の型定義は次のとおりである。

```
RFC 3852[5] 5.2 節 EncapsulatedContentInfo Type より
EncapsulatedContentInfo ::= SEQUENCE {
  eContentType ContentType,
  eContent [0] EXPLICIT OCTET STRING OPTIONAL }
```

eContentType には署名対象文書(データ)のデータ型を指定し、eContent には署名対象文書(データ)のデータバイト列を格納する。署名対象範囲を正確に記すと、eContent の値となる OCTET STRING 型データの値である(MUST[10]5.4)。

eContentType には通常 id-data を指定する。id-data の OID は次のとおりである。

```
RFC 3852[5] 4 節 Data Content Type より
id-data OBJECT IDENTIFIER ::= { iso(1) member-body(2)
                                us(840) rsadsi(113549) pkcs(1) pkcs7(7) 1 }
```

署名対象文書を CMS に含めない場合には、eContentType に id-data を指定して eContent フィールドを省略する。この場合、署名対象文書と CMS や CAdES などの署名データとの関連はアプリケーション側で管理することとなる。

コラム：タイムスタンプトークンの eContent である TSTInfo について

ちなみに、狭義の CAdES は CMS SignedData であり、RFC 3161[5]準拠のタイムスタンプトークンもまた CMS SignedData である。タイムスタンプトークンの場合は、eContentType に id-ct-TSTInfo を指定し、eContent には TSTInfo 型のデータを格納する。

```
RFC 3161[5] 2.4.2 節 Response Format より
id-ct-TSTInfo  OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) ct(1) 4}
```

```
RFC 3161[5] 2.4.2 節 Response Format より
TSTInfo ::= SEQUENCE  {
  version                      INTEGER  { v1(1) },
  policy                       TSAPolicyId,
  messageImprint               MessageImprint,
  -- MUST have the same value as the similar field in TimeStampReq
  serialNumber                 INTEGER,
  -- Time-Stamping users MUST be ready to accommodate integers
```

```
  -- up to 160 bits
  genTime                 GeneralizedTime,
  accuracy                Accuracy                OPTIONAL,
  ordering                BOOLEAN             DEFAULT FALSE,
  nonce                   INTEGER                 OPTIONAL,
  -- MUST be present if the similar field was present
  -- in TimeStampReq.  In that case it MUST have the same value.
  tsa                     [0] GeneralName         OPTIONAL,
  extensions              [1] IMPLICIT Extensions   OPTIONAL  }
```

1.1.3.4. certificates フィールド(CertificateSet 型)：証明書群

certificates フィールドは署名者の証明書、属性証明書またトラストアンカまでの証明書チェーンの全てまたは一部を格納することができるオプショナルフィールドである。

CertificateSet の型定義は次の通りである。

```
RFC 3852[5] 10.2.3 節 CertificateSet より
CertificateSet ::= SET OF CertificateChoices


RFC 3852[5] 10.2.2 節 CertificateChoices より
CertificateChoices ::= CHOICE {
  certificate Certificate,
  extendedCertificate [0] IMPLICIT ExtendedCertificate,  -- Obsolete
  v1AttrCert [1] IMPLICIT AttributeCertificateV1,        -- Obsolete
  v2AttrCert [2] IMPLICIT AttributeCertificateV2,
  other [3] IMPLICIT OtherCertificateFormat }
```

certificates には任意の証明書を格納できるようになっているが、一般的には X.509 公開鍵証明書を格納する。その場合 CertificateChoices では certificate を選択する。

Certificate の型定義は次のとおりである。

```
    RFC 3280[6] 4.1 節 Basic Certificate Fields より
RFC 3280[6] 4.1 節 Basic Certificate Fields より
Certificate  ::=  SEQUENCE  {
  tbsCertificate       TBSCertificate,
  signatureAlgorithm   AlgorithmIdentifier,
  signatureValue       BIT STRING  }
TBSCertificate  ::=  SEQUENCE  {
```

```
version         [0]  EXPLICIT Version DEFAULT v1,
serialNumber         CertificateSerialNumber,
signature            AlgorithmIdentifier,
issuer               Name,
validity             Validity,
subject              Name,
subjectPublicKeyInfo SubjectPublicKeyInfo,
issuerUniqueID  [1]  IMPLICIT UniqueIdentifier OPTIONAL,
                     -- If present, version MUST be v2 or v3
subjectUniqueID [2]  IMPLICIT UniqueIdentifier OPTIONAL,
                     -- If present, version MUST be v2 or v3
extensions      [3]  EXPLICIT Extensions OPTIONAL
                     -- If present, version MUST be v3 }
```

公開鍵証明書はルート証明書、もしくは署名付き文書の受け取り手が信頼している証明書から署名者の証明書までの一連の証明書を全て格納することができる。(この一連の証明書を証明書チェーンと呼ぶ。)署名者が複数の場合、即ち後述の SignerInfo を複数持つ場合には、署名者の数に応じて署名検証に必要とされる全ての証明書チェーンを全て格納することもできる。署名検証に必要とされない証明書が余分に入っていても構わない。CAdES フォーマットで長期保存する場合には、本フィールドを用いるよりも後述の CAdES-X-Long フォーマットにより非署名属性に証明書チェーンや失効情報を格納する。

certificates には署名の目的に応じて属性証明書を格納することもできる。その場合 CertificateChoices では v2AttrCert を選択する([10]5.1)。

1.1.3.5. crls フィールド(CertificateRevocationLists 型):証明書失効情報

本フィールドでは、署名者証明書並びにその証明書チェーンの検証に必要な失効情報を格納することができるオプショナルフィールドである。RevocationInfoChoices の型定義は次の通りである。

```
RFC 3852[5] 10.2.1 節 RevocationInfoChoices より
RevocationInfoChoices ::= SET OF RevocationInfoChoice
RevocationInfoChoice ::= CHOICE {
        crl CertificateList,
        other [1] IMPLICIT OtherRevocationInfoFormat }
```

一般に CRL(証明書失効リスト)を格納するが、other フィールドを利用することで、証明書失効情報として OCSP レスポンスなど他の任意の失効情報を格納できる可能性があるが、

そのためには、格納の種類を示すオブジェクト識別子を別途規定しなければならない。CAdES フォーマットで長期保存する場合には、本フィールドを用いるよりも後述の CAdES-X-Long フォーマットにより非署名属性に CRL や OCSP レスポンスを格納する。

1.1.3.6. SignerInfo 型

一つの SignerInfo に一人の署名者の署名情報を格納されている。SingerInfo の型定義は次のとおりである。

```
RFC 3852[5] 5.3 節 SignerInfo Type より
SignerInfo ::= SEQUENCE {
  version            CMSVersion,
  sid                SignerIdentifier,
  digestAlgorithm    DigestAlgorithmIdentifier,
  signedAttrs        [0] IMPLICIT SignedAttributes OPTIONAL,
  signatureAlgorithm SignatureAlgorithmIdentifier,
  signature          SignatureValue,
  unsignedAttrs      [1] IMPLICIT UnsignedAttributes OPTIONAL }
SignedAttributes ::= SET SIZE (1..MAX) OF Attribute
UnsignedAttributes ::= SET SIZE (1..MAX) OF Attribute
Attribute ::= SEQUENCE {
  attrType        OBJECT IDENTIFIER,
  attrValues      SET OF AttributeValue }
AttributeValue ::= ANY
```

各フィールドの型及び値は次のとおりである。

1.1.3.7. version フィールド(CMSVersion 型)：SignerInfo の版番号

本 version フィールドは SignerInfo 構造を表す版番号を値に持つ。取り得る値は以下の通りである。[10]

- sid フィールドの値が issuerAndSerialNumber ならば 1
- sid フィールドの値が subjectKeyIdentifier ならば 3

1.1.3.8. sid フィールド(SignerIdentifier 型)：署名者識別子

sid フィールドは、署名者の公開鍵証明書を識別するための識別情報を保持する。CAdES では、より厳密な署名者情報を持つ SigningCertificate 属性が必須であるため(MUST[18]5.7)、本フィールドの検証は、あまり重要ではない。

SignerIdentifier の型定義は次のとおりである。

```
RFC 3852[5] 5.3 節 SignerInfo type より
SignerIdentifier ::= CHOICE {
  issuerAndSerialNumber  IssuerAndSerialNumber,
  subjectKeyIdentifier   [0] SubjectKeyIdentifier }
IssuerAndSerialNumber ::= SEQUENCE {
  issuer Name,
  serialNumber CertificateSerialNumber }
CertificateSerialNumber  ::=  INTEGER
SubjectKeyIdentifier ::= KeyIdentifier
KeyIdentifier ::= OCTET STRING
```

CAdES v1.7.3 以降では、sid フィールドの値について issuerAndSerialNumber と subjectKeyIdentifier のいずれを使用しても構わないが、CAdES v.1.6.3 以前では、多くの場合 subjectKeyIdentifier を使用する必要があった(コラム参照)。

コラム：CMS バージョンと sid フィールドの値について

CAdES v1.6.3 以前では、CMS バージョン 3 の SignedData でなければならない(SHALL[17]8.1)としていたが、ECOM からの継続的な提言により、この制限が撤廃された。CMS の定義[10]により、CMS バージョン 3 の SignedData であるためには、現在、非推奨となっている属性証明書 V1 を certificates フィールドに含めるか、SignerInfo のバージョンをバージョン 3、即ち sid フィールドに subjectKeyIdentifier を保持する必要がある。

CAdES v1.7.3 以降、この制限が撤廃されたことにより、subjetKeyIdentifier 拡張を持たない署名者証明書であっても問題なく利用できるようになっている。

1.1.3.9. digestAlgorithm フィールド(DigestAlgorithmIdentifier 型)：使用するハッシュアルゴリズム

本フィールドはある署名者が署名対象文書に対して適用するハッシュアルゴリズムの OID を格納する。

DigestAlgorithmIdentifier の型定義は次のとおりである。

```
RFC 3852[5] 10.1.1 節 DigestAlgorithmIdentifier より
DigestAlgorithmIdentifier ::= AlgorithmIdentifier

RFC 3280[6] 4.1.1.2 節 signatureAlgorithm より
```

```
AlgorithmIdentifier  ::=  SEQUENCE  {
     algorithm             OBJECT IDENTIFIER,
     parameters            ANY DEFINED BY algorithm OPTIONAL  }
```

よく利用されるハッシュアルゴリズムの OID をに掲載する。

(id-md5 は[27]C. ASN.1 MODULE より、それ以外は[9]より)

表 1.1.3.9.1: 一般的なハッシュアルゴリズム

アルゴリズム名	OID
MD5	id-md5 OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) digestAlgorithm(2) 5 }
SHA-1	id-sha1 OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) oiw(14) secsig(3) algorithm(2) 26 }
SHA-256	id-sha256 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistalgorithm(4) hashalgs(2) 1 }
SHA-384	id-sha384 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistalgorithm(4) hashalgs(2) 2 }
SHA-512	id-sha512 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistalgorithm(4) hashalgs(2) 3 }

上記アルゴリズムに対しては parameters フィールドを省略するのが正しいが、事実上、多くの実装において parameters フィールドに NULL を設定しているため、双方を受理できるように実装する必要がある。

1.1.3.10. signedAttrs フィールド(SignedAttributes 型) : 署名対象属性

本フィールドは署名対象となる属性の集まりである。

表 1.1.3.14.1 に CAdES で使用される署名属性、および各属性の掲載ドキュメントを示す。

表 1.1.3.10.1: CAdES の署名属性一覧

	必須	オプション
CMS 属性 [RFC3852]	Content Type Message Digest	Signing Time
ESS 属性 [RFC2634]	Signing Certificate	Content Reference Content Identifier Contet Hints
その他の属性	Signature policy identifier	Commitment type indication

[ETSI TS 101 733]	(CAdES-EPES の場合)	Signer location Signer attributes Content time-stamp

CMS SigndData では signedAttrs はオプショナルフィールドだが、CAdES では必須である。encapContentInfo フィールドの eContentType フィールドに id-data 以外のデータ型を指定する場合には SignerInfo に含めなくてはならない(MUST[10]5.3)。また、SigningCertificate 属性が必須であるため、signedAttrs フィールドは必須であり、ContentType 属性と MessageDigest 属性が必須となる。

signedAttrs フィールドを省略しない場合、signedAttrs フィールドには少なくとも Content Type 属性と Message Digest 属性の 2 属性を必ず持たせなければならない(MUST[10]5.3)。それぞれの属性の詳細については後述する。

CAdES では signedAttrs を省略することはできず、Content Type 属性と Message Digest 属性に、後述する Signing Certificate 属性を加えた少なくとも 3 つの属性を必ず持たせなければならない。(MUST[18]5.7)

1.1.3.11. signatureAlgorithm フィールド(SignatureAlgorithmIdentifier 型):使用する署名アルゴリズム

本フィールドは、CAdES では必須となる signedAttrs フィールドに対する署名値である signature フィールドを計算するのに使用した署名アルゴリズムを指定する。

SignatureAlgorithmIdentifier の型定義は次のとおりである。

```
SignatureAlgorithmIdentifier ::= AlgorithmIdentifier

AlgorithmIdentifier  ::=  SEQUENCE  {
     algorithm               OBJECT IDENTIFIER,
     parameters              ANY DEFINED BY algorithm OPTIONAL  }
```

一般的な署名アルゴリズムの OID を表 1.1.3.11.1 に示す。([27]C. ASN.1 MODULE より)

表 1.1.3.11.1: 一般的な署名アルゴリズム

アルゴリズム名	OID
<digest>withRSA	rsaEncryption OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 1 }
MD5WithRSA	md5WithRSAEncryption OBJECT IDENTIFIER ::= {iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 4}
SHA1WithRSA	sha1WithRSAEncryption OBJECT IDENTIFIER ::=

	{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 5 }
SHA256WithRSA	sha256WithRSAEncryption OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 11 }
SHA384WithRSA	sha384WithRSAEncryption OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 12 }
SHA512WithRSA	sha512WithRSAEncryption OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 13 }

上記アルゴリズムに対しては parameters フィールドを省略するのが正しいが、事実上、多くの実装において parameters フィールドに NULL を設定しているため、双方を受理できるように実装する必要がある。

1.1.3.12. signature フィールド(SignatureValue 型)：署名値

本フィールドは一人の署名者の署名値オクテットを格納する。SignatureValue の型定義は次の通りである。

```
RFC 3852[5] 5.3 節 SignerInfo Type より
SignatureValue ::= OCTET STRING
```

1.1.3.13. unsignedAttrs フィールド(UnsignedAttributes 型)：非署名属性群

本フィールドは署名対象としない属性の集まりである。後に本フィールドに含まれる属性を追加、変更したとしても、署名者署名の改竄とはならない。狭義の CAdES フォーマットでは非署名属性に含まれるのは CounterSignature 属性のみである。

1.1.3.14. Content Type 属性：署名対象データのデータ型(必須)

本属性は署名対象データのデータ型の OID を格納しておく属性である。

本属性は signedAttrs フィールドが存在する場合には必ず含めなければならない(MUST[10]11.1)。

attrType には次の OID を格納する。

```
RFC 3852[5] 11.1 節 Content Type より
id-contentType OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) 3 }
```

attrValues には encapContentInfo フィールドの eContentType フィールドで指定したの

と同じデータ型の OID を格納する(MUST[10]11.1)。

attrValues の値は次に示すデータ構造をしている。

```
RFC 3852[5] 11.1 節 Content Type より
ContentType ::= OBJECT IDENTIFIER
```

属性の性質上 attrValues は必ず１つのみ値を持つものであり、attrValues は値を持たないこと、もしくは２つ以上の値を値を持つことがあってはならない。また SignerInfo に本属性のインスタンスを複数持たせてはならない(MUST NOT[10]11.1)。

1.1.3.15. Message Digest 属性：署名対象データのハッシュ値(必須)

本属性は署名対象データのハッシュ値を格納しておく属性である。本属性は signedAttrs フィールドが存在する場合には必ず含めなければならない(MUST[10]11.2)。

attrType には次の OID を格納する。

```
RFC 3852[5] 11.2 節 Message Digest より
id-messageDigest OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) 4 }
```

attrValues には場合に応じてそれぞれ以下のデータを格納する。

(1) encapContentInfo の eContent フィールドが存在する場合

eContent の値である OCTET STRING 型データの値に digestAlgorithm で指定されるハッシュアルゴリズムを適用して得られるハッシュデータを格納する。

(2) encapContentInfo の eContent フィールドが存在しない場合

署名フォーマットの上位のデータフォーマット、もしくは文書フォーマット内で管理される署名対象データに digestAlgorithm で指定されるハッシュアルゴリズムを適用して得られるハッシュデータを格納する。

attrValues の値は次に示すデータ構造をしている。

```
RFC 3852[5] 11.2 節 Message Digest より
MessageDigest ::= OCTET STRING
```

属性の性質上 attrValues は必ず１つのみ値を持つものであり、attrValues は値を持たないこと、もしくは２つ以上の値を持つことがあってはならない。また SignerInfo に本属性のインスタンスを複数持たせてはならない(MUST NOT[10]11.2)。

1.1.3.16. Signing Certificate 属性：署名者証明書参照情報(必須)

本属性は署名者証明書を識別ために署名者証明書のハッシュ値、証明書発行者識別名、証明書シリアル番号などを格納するための属性である。

Signing Certificate 属性は公開鍵証明書に適用するハッシュアルゴリズムに応じて ESS signing certificate 属性となるか、ESS signing certificate v2 属性となるか変わる。ハッシュアルゴリズムが SHA1 の場合は ESS signing certificate 属性となり、ハッシュアルゴリズムが SHA1 以外の場合は ESS signing certificate v2 属性となる。

ESS signing certifcate 属性

attrType には次の OID を格納する。

```
RFC 2634[2] 5.4 節 Signing Certificate Attribute Definition より
id-aa-signingCertificate OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16)
id-aa(2) 12 }
```

attrValues の各値は次に示すデータ構造をしている。

```
RFC 2634[2] 5.4 節 Signing Certificate Attribute Definition より
SigningCertificate ::=  SEQUENCE {
  certs        SEQUENCE OF ESSCertID,
  policies     SEQUENCE OF PolicyInformation OPTIONAL  -- 使用しない }
ESSCertID ::=  SEQUENCE {
  certHash                 Hash,
  issuerSerial             IssuerSerial OPTIONAL }
Hash ::= OCTET STRING -- SHA1 hash of entire certificate
IssuerSerial ::= SEQUENCE {
  issuer                   GeneralNames,
  serialNumber             CertificateSerialNumber }
```

ESS signing certificate v2 属性

attrType には次の OID を格納する。

```
Internet Draft ESS Update[14]: Adding CertID Algorithm Agility
5.4.1 節 Signing Certificate Attribute Definition Version 2 より
id-aa-signingCertificateV2 OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16) id-aa(2) 47 }
```

attrValues の各値のデータ構造は次の通りである。

```
Internet Draft ESS Update[14]: Adding CertID Algorithm Agility
5.4.1 節 Signing Certificate Attribute Definition Version 2 より
SigningCertificateV2 ::=  SEQUENCE {
     certs        SEQUENCE OF ESSCertIDv2,
     policies     SEQUENCE OF PolicyInformation OPTIONAL -- 使用しない }
ESSCertIDv2 ::=  SEQUENCE {
        hashAlg                 AlgorithmIdentifier DEFAULT {id-sha256},
        certHash                Hash,
        issuerSerial            IssuerSerial OPTIONAL }
Hash ::= OCTET STRING
IssuerSerial ::= SEQUENCE {
        issuer                  GeneralNames,
        serialNumber            CertificateSerialNumber }
```

平成 17 年度の CMS 長期署名プロファイル(Version 1.0)では Signing Certificate 属性内で使用するハッシュアルゴリズムが SHA1 以外の場合、Signing Certificate 属性は Other signing certificate 属性になるとしていた。しかし今後は同じ機能を提供しながらも、よりシンプルな構造である ESS signing certificate v2 属性を使用することとする。

参考として Other signing certificate 属性の OID とデータ型定義も以下に掲載する。

OID

```
ETSI TS 101 733 V1.7.3[18] 5.7.3.3 節 Other signing certificate attribute definition より
id-aa-ets-otherSigCert OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16) id-aa(2) 19 }
```

データ型定義

```
    ETSI TS 101 733 V1.7.3[18] 5.7.3.3 節 Other signing certificate attribute definition より
    OtherSigningCertificate ::=  SEQUENCE {
      certs           SEQUENCE OF OtherCertID,
      policies        SEQUENCE OF PolicyInformation OPTIONAL -- 使用しない }
    OtherCertID ::=  SEQUENCE {
      otherCertHash   OtherHash,
      issuerSerial    IssuerSerial OPTIONAL  -- OPTIONAL }
    OtherHash ::= CHOICE {
```

```
    sha1Hash          OtherHashValue,   -- SHA-1 ハッシュ
    otherHash         OtherHashAlgAndValue }
OtherHashValue ::= OCTET STRING
OtherHashAlgAndValue ::= SEQUENCE {
    hashAlgorithm     AlgorithmIdentifier,
    hashValue         OtherHashValue  }
```

ESS signing certificate 属性と ESS signing certificate v2 属性の共通事項

certs フィールドに公開鍵証明書のハッシュ値を複数格納する場合、その SEQUENCE の並びの最初のハッシュ値は署名者証明書のハッシュ値でなければならない(MUST[2]5.4)。

certs フィールドに格納するハッシュ値のハッシュ対象は公開鍵証明書データ全体である(証明書に付与される署名データも含む)。

署名検証時に署名者証明書だと特定された公開鍵証明書のハッシュ値が certs フィールドに収められた最初のハッシュ値と一致しない場合、署名は無効だと判定されなくてはならない(SHALL[18]5.7.3)。

issuerSerial は OPTION と定義されているが、必ず使用するものとする(SHALL[18]5.7.3)。また SignerInfo の sid が IssuerAndSerialNumber 型のデータを取る場合、issuerSerial が取る発行者名とシリアルナンバーは sid のものと同じでなければならない(SHALL[18]5.7.3)。

本属性が存在する場合、attrValues は値を必ず 1 つのみ持っていなくてはならない。(値を持たないこと、または 2 つ以上の値を値を持つことがあってはならない。)また SignerInfo に本属性のインスタンスを複数持たせてはならない(MUST NOT[2]5.4)。

コラム：sid に対する攻撃について

本属性は署名者証明書を特定するという目的においては sid と同じであるが、sid は署名保護されるフィールドではないため、sid が改ざんされてもその事実を署名検証時に検知することができないという脆弱性を補うための属性である。

sid の脆弱性を突いた攻撃は sid と同様に署名保護されない SignerInfo の certificates フィールドの脆弱性を同時に突くことで成されると考えられる。攻撃者は図 1.1.3.16.1 に示すように sid を改ざんすることで署名検証者に偽の署名者証明書を用いて署名検証させるこ

とができるようになる。偽の署名者証明書には証明書に含まれる公開鍵が(a)偽の公開鍵の場合と、(b)本物の署名者公開鍵の場合の２通りの場合がある。(a)の場合は署名検証者に常に署名検証を失敗させるという一種の DoS 攻撃(Denial of Service attack)である。署名検証を失敗させるという DoS 攻撃は sid の脆弱性を突かなくとも可能で、その DoS 攻撃自体を完全に防ぐ事は不可能であり、また(a)に関しては攻撃の検知も出来るため(a)の攻撃は問題があまり大きくない。(b)の場合は偽の署名者証明書を用いた署名検証が成功してしまい、攻撃が検知できないため問題が大きい。本属性は(a)と(b)の両方の場合に対処することが出来る。

署名者は署名時に署名者証明書のハッシュ値を本属性に格納、また本属性を署名対象として署名保護しておき、署名検証者は署名者証明書だと特定した証明書のハッシュ値と本属性に格納されているハッシュ値が一致するか確認することで、署名検証者は偽の署名者証明書で署名検証させようとする攻撃を検知することができる。

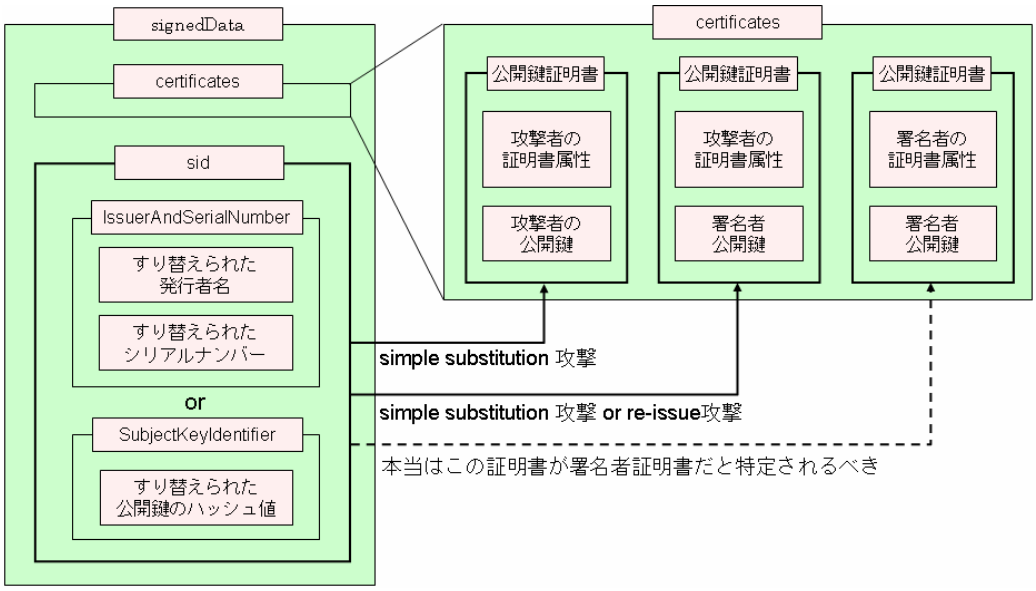


図 1.1.3.16.1: sid の脆弱性

(a)の攻撃、及び(b)の攻撃の内、エンドユーザが署名者公開鍵に元の署名者証明書とは異なる証明書属性で発行された証明書を手に入れて元の署名者証明書とすり替える攻撃を “simple substitution 攻撃”という。(b)の攻撃の内、認証局が同じ公開鍵に対してルート証明書などを再発行、特にその際に証明書ポリシを変えてしまうなどをすることを “re-issue 攻撃”という。(b)の攻撃は同じ公開鍵に対しては証明書を再発行しないようにする、という認証局の運用によっても防ぐことは可能であるが、本属性は署名者と署名検証者が気をつけることで(b)の攻撃を防げるよう署名データフォーマットに機能を加えるも

のである([2]5.1)。

1.1.3.17. Signature policy identifier 属性：署名ポリシ識定子(CAdES-EPES の場合に必須)

本属性は署名ポリシを特定するための情報(署名ポリシID、署名ポリシへのリファレンス)を格納する属性である。

署名ポリシは、署名者と署名の検証者が守るべき一連の規則である。署名者は自分の意図と合致する署名ポリシを取得して署名に含め、検証者は検証処理の一環として署名ポリシを参照することにより、署名の意味や有効性を確認する。署名ポリシの記述形式には、人間が読んで理解できるフリーテキストによる形式と、ASN.1 や XML によるコンピュータによる処理が可能な形式とがある。署名ポリシには、次のような効果が期待される([43])。

(1)署名の意図や意味を明示し、電子署名の安全性を強化する。
(2)署名処理系の汎用利用が可能となる。
(3)長期署名の証明力を向上する。

図 1.1.3.18.1 に署名ポリシの構造を示す。

```
RFC 3125[3] 3.1節 Overall ASN.1 Structureより
SignaturePolicy ::= SEQUENCE {
        signPolicyHashAlg       AlgorithmIdentifier,
        signPolicyInfo          SignPolicyInfo,
        signPolicyHash          SignPolicyHash      OPTIONAL }
SignPolicyHash ::= OCTET STRING
SignPolicyInfo ::= SEQUENCE {
        signPolicyIdentifier            SignPolicyId,
        dateOfIssue                     GeneralizedTime,
        policyIssuerName                PolicyIssuerName,
        fieldOfApplication              FieldOfApplication,
        signatureValidationPolicy       SignatureValidationPolicy,
        signPolExtensions               SignPolExtensions    OPTIONAL  }
SignPolicyId ::= OBJECT IDENTIFIER
PolicyIssuerName ::= GeneralNames
FieldOfApplication ::= DirectoryString

RFC 3125[3] 3.2節 Signature Validation Policyより
SignatureValidationPolicy ::= SEQUENCE {
        signingPeriod           SigningPeriod,
        commonRules             CommonRules,
        commitmentRules         CommitmentRules,
        signPolExtensions       SignPolExtensions  OPTIONAL  }
SigningPeriod ::= SEQUENCE {
            notBefore       GeneralizedTime,
            notAfter        GeneralizedTime OPTIONAL }
```

図 1.1.3.18.1: 署名ポリシの構造

署名ポリシに関する詳細については[43]を参照のこと。

attrType には次の OID を格納する。

```
ETSI TS 101 733 V1.7.3[18] 5.8.1節 Signature policy identifierより
id-aa-ets-sigPolicyId OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16)
id-aa(2) 15 }
```

attrValues の各値は次に示すデータ構造をしている。

```
ETSI TS 101 733 V1.7.3[18] 5.8.1節 Signature policy identifier より
SignaturePolicyIdentifier ::= CHOICE {
          SignaturePolicyId        SignaturePolicyId,
          SignaturePolicyImplied   SignaturePolicyImplied -- 使用しない }
SignaturePolicyId ::= SEQUENCE {
          sigPolicyIdentifier      SigPolicyId,
          sigPolicyHash            SigPolicyHash,
          sigPolicyQualifiers      SEQUENCE SIZE (1..MAX) OF
                                   SigPolicyQualifierInfo    OPTIONAL }
SignaturePolicyImplied ::= NULL
SigPolicyId ::= OBJECT IDENTIFIER
SigPolicyHash ::= OtherHashAlgAndValue
OtherHashAlgAndValue ::= SEQUENCE {
          hashAlgorithm    AlgorithmIdentifier,
          hashValue        OtherHashValue }
OtherHashValue ::= OCTET STRING
SigPolicyQualifierInfo ::= SEQUENCE {
          sigPolicyQualifierId    SigPolicyQualifierId,
          sigQualifier            ANY DEFINED BY sigPolicyQualifierId }
SigPolicyQualifierId ::= OBJECT IDENTIFIER
id-spq-ets-uri OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16)
  id-spq(5) 1 }
SPuri ::= IA5String
id-spq-ets-unotice OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16)
  id-spq(5) 2 }
SPUserNotice ::= SEQUENCE {
          noticeRef           NoticeReference OPTIONAL,
          explicitText        DisplayText OPTIONAL }
NoticeReference ::= SEQUENCE {
          organization        DisplayText,
          noticeNumbers       SEQUENCE OF INTEGER }
DisplayText ::= CHOICE {
          visibleString       VisibleString (SIZE (1..200)),
```

```
        bmpString           MBPString     (SIZE (1..200)),
        utf8String          UTF8String    (SIZE (1..200))  }
```

1.1.3.18. Signing Time 属性：確かな信頼性はない署名時刻(オプション)

本属性は署名がなされた時刻を格納しておく属性である。

本属性に格納される時刻に正確さは要求されない。また署名者が自らの PC の時刻を変更して署名するなどして操作可能な時刻であるため確かな信頼性はない時刻である。本属性の時刻を信頼するか否かは署名検証者の判断に委ねられ、予め信頼関係が築かれている間柄でのみ通用する時刻である。時刻認証された信頼性のある署名時刻を格納したい場合は後述の CAdES-T フォーマットを用いる。

attrType には次の OID を格納する。

```
RFC 3852[5] 11.3 節 Signing Time より
id-signingTime OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) 5 }
```

attrValues には UTCTime 型の時刻データか GeneralizedTime 型の時刻データを格納する。1950 年 1 月 1 日から 2049 年 12 月 31 日までの日時を表現する場合は UTCTime 型の時刻データでなければならない。1950 年以前、もしくは 2049 年以後の日時を表現する場合は GeneralizedTime 型の時刻データでなければならない(MUST[10]11.3)。

UTCTime 型の時刻は協定世界時で表現されなければならない。そして(たとえ値が 0 であっても)秒までのデータを含めなければならない。(つまり時刻データは YYMMDDHHMMSSZ となる。)真夜中は “YYMMDD000000Z” と表現される(MUST[10]11.3)。

GeneralizedTime 型の時刻も同じく協定世界時で表現し、秒までのデータを含めなければならない。(つまり時刻データは YYYYMMDDHHMMSSZ となる。)GeneralizedTime 型の時刻にミリ秒以下のデータを含めてはならない(MUST NOT[10]11.3)。
attrValues の値は次に示すデータ構造をしている。

```
RFC 3852[5] 11.3 節 Signing Time より
SigningTime ::= Time
Time ::= CHOICE {
  utcTime UTCTime,
  generalizedTime GeneralizedTime }
```

属性の性質上 attrValues は必ず 1 つのみ値を持つものであり、attrValues は値を持たないこと、もしくは 2 つ以上の値を値を持つことがあってはならない。また SignerInfo に本属性のインスタンスを複数持たせてはならない(MUST NOT[10]11.3)。

1.1.3.19. Content Reference 属性：他の署名文書への参照情報(オプション)

本属性は他の署名文書(signedData)の識別子を参照情報として格納する属性である。

本属性は 2 つの署名文書がお互いに何らかの関係がある際に、片方の文書からもう片方の文書へ信頼性の高い参照をしたい場合に用いられる。図 1.1.3.19.1 を例にとると、本属性を使用することで、署名文書Ｂが参考にした文書は確かに署名文書Ａであることを第三者が確認できるようになる。

署名文書から署名文書への信頼性の高い参照を実現するには参照元と参照先の署名文書それぞれに細工をする必要がある。具体的には予め参照先の署名文書に 5-7 で後述する Content Identifier 属性を含めておき(ステップ 1)、続いて参照元の署名文書に Content Reference 属性を含める(ステップ 2)。この 2 ステップを踏むことで信頼性の高い参照が実現できる。つまり、他の署名文書から参照される可能性のある文書に署名をする際には Content Identifier 属性を含めておくことが望ましい。

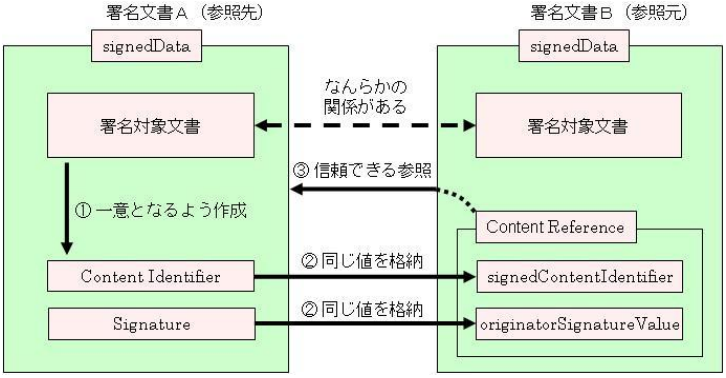


図 1.1.3.19.1: Content Reference 属性の用法

attrType には次の OID を格納する。

```
RFC 2634[2] 2.11 節 Signed Content Reference Attribute より
id-aa-contentReference   OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16)
  id-aa(2) 10 }
```

attrValues の各値は次に示すデータ構造をしている。

```
RFC 2634[2] 2.11 節 Signed Content Reference Attribute より
ContentReference ::= SEQUENCE {
    contentType ContentType,
    signedContentIdentifier ContentIdentifier,
    originatorSignatureValue OCTET STRING }
```

ContentReference 属性の contentType フィールドには参照先の署名文書の eContentType フィールドをそのままを格納し、同じく signedContentIdentifier フィールドには参照先の署名文書の Content Identifier 属性をそのまま格納し、同じく originatorSignatureValue フィールドには参照先の署名文書の Signature フィールドをそのまま格納する。

1.1.3.20. Content Identifier 属性：コンテント識別子(オプション)

本属性は署名対象文書であるコンテントを一意に特定する情報を格納する属性である。

ユースケースは前節で前述したように、署名文書間に信頼性の高いリンクを貼る用途が想定されている。本書内で Content Identifier が登場する場合、コンテントとは署名対象文書である。署名対象文書から Content Identifier の具体的な作成方法は RFC 2634[2]、ETSI TS 101 733 V1.7.3[18]などには記載されていないが、署名対象文書のハッシュ値を Content Identifier とすることが一般的だと考えられる。またコンテントの一意性を確保するために、署名対象文書のハッシュ値などの Content Identifier にユーザ名・公開鍵データ・時刻情報・乱数などを連結させて新たに Content Identifier とすることが RFC 2634[2]、ETSI TS 101 733 V1.7.3[18]で推奨されている。

attrType には次の OID を格納する。

```
RFC 2634[2] 2.7 節 Receipt Request Syntax より
id-aa-contentIdentifier OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16)
id-aa(2) 7 }
```

attrValues の各値は次に示すデータ構造をしている。

```
RFC 2634[2] 2.7 節 Receipt Request Syntax より
ContentIdentifier ::= OCTET STRING
```

1.1.3.21. Contet Hints 属性：署名対象文書フォーマットの補足情報(オプション)

本属性は署名対象文書のデータフォーマットに関する補足情報を格納する属性である。

署名対象文書のデータフォーマットの型は基本的に eContentType フィールドに格納されているが、eContentType フィールドは OID を格納するフィールドなので、署名対象文書が OID を割り振られていないデータフォーマットである場合、eContentType では署名対象文書のデータフォーマットを十分に表現することができない(その場合、eContentType には id-data が格納されているだろう)。本属性はテキスト情報を内包するので、こうした場合に本属性を用いることによって、署名対象文書のデータフォーマットに関してより正確な情報を提供することができる。また eContentType フィールドに格納される OID が id-data 以外の十分に意味のあるデータフォーマットを示している場合でも、本属性を併用することで、署名対象文書のデータフォーマットに関してより詳細な情報を提供することができる。

attrType には次の OID を格納する。

```
RFC 2634[2] 2.9 節 Content Hints より
id-aa-contentHint OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16)
id-aa(2) 4}
```

attrValues の各値は次に示すデータ構造をしている。

```
RFC 2634[2] 2.9 節 Content Hints より
ContentHints ::= SEQUENCE {
    contentDescription UTF8String (SIZE (1..MAX)) OPTIONAL,
    contentType ContentType }
```

ContentHints の contentType フィールドには eContentType フィールドをそのまま格納する。

ETSI TS 101 733 V1.7.3[18]は署名対象文書(データ)が MIME タイプで指定可能なデータフォーマットである場合を Content Hints 属性の用法例として出している。その場合には以下のようにして Content Hints 属性を使用する。

- contentType には id-data を指定する。
- contentDescription には MIME タイプを格納する。

MIME のタイプの例としては、text/html、image/gif、audio/mp3、video/mpeg4、

application/pdf などが挙げられる。署名文書(signdData)の処理系は、contentDescrpition に格納された MIME タイプを参照することで、内包する署名対象文書(eContent)を適切な処理系に渡すことができる(図 1.1.3.21.1 参照)。

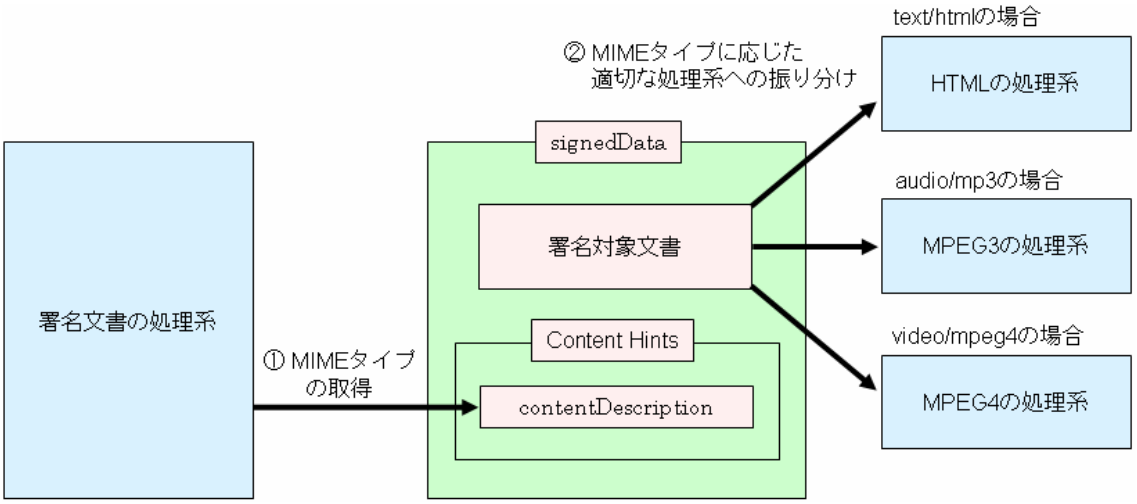


図 1.1.3.21.1: Content Hints 属性の用法例

1.1.3.22. Commitment type indication 属性：署名意図の表明(オプション)

本属性は署名者がどのような意図で署名したかを表明するために利用される。

署名意図は通常いくつかの決まりきった意図であることが多いため、典型的な署名意図を予めいくつか用意しておき、その中のいずれかを署名時に選択して署名意図を表明することが一般的になると予想される。

実際の運用に当たっては貿易機関や立法機関などを署名意図の登録機関とし、予め用意しておいた署名意図を登録機関に登録しておき、署名者と署名の受け手の両者間で署名意図の解釈を合意しておくことが適切であると考えられる([18]5.11.1)。

登録機関に登録されていない署名意図で署名する場合、そのような典型的ではない署名意図を別途定義して表明することも可能である。署名ポリシは署名意図(Commitment)定義のためのフィールドを持っており、署名意図の定義は署名ポリシ内で行うことができる。

署名意図定義のデータ構造を次に示す。

```
RFC 3125[3] 3.4 節 Commitment Rules より
CommitmentType ::= SEQUENCE {
        identifier                      CommitmentTypeIdentifier,
```

```
        fieldOfApplication          [0] FieldOfApplication OPTIONAL,
        semantics                   [1] DirectoryString OPTIONAL }
```

attrType には次の OID を格納する。

```
ETSI TS 101 733 V1.7.3[18] 5.11.1節 Commitment type indication attributeより
id-aa-ets-commitmentType OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16)
  id-aa(2) 16 }
```

attrValues の各値は次に示すデータ構造をしている。

```
ETSI TS 101 733 V1.7.3[18] 5.11.1節 Commitment type indication attributeより
CommitmentTypeIndication ::= SEQUENCE {
    commitmentTypeId CommitmentTypeIdentifier,
    commitmentTypeQualifier SEQUENCE SIZE (1..MAX)
OF CommitmentTypeQualifier OPTIONAL }

      CommitmentTypeIdentifier ::= OBJECT IDENTIFIER

      CommitmentTypeQualifier ::= SEQUENCE {
        commitmentTypeIdentifier CommitmentTypeIdentifier,
        qualifier  ANY DEFINED BY commitmentTypeIdentifier }
```

[18]で定義されている 6 つの典型的な署名意図を表 1.1.3.22.1 に示す。

表 1.1.3.22.1: 典型的な署名意図

署名意図名	OID
	署名意図
Proof of origin	id-cti-ets-proofOfOrigin OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) cti(6) 1 }
	署名者がその文書を生成したこと、承認したこと、そして送信したことを示す。
Proof of receipt	id-cti-ets-proofOfReceipt OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) cti(6) 2 }
	署名者がその文書を受け取ったことを示す。

Proof of delivery	id-cti-ets-proofOfDelivery OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) cti(6) 3 }
	署名をした TSP(信頼できるサービスプロバイダ)が文書を TSP のローカルに置き、文書受信者がその文書にアクセスできる状態にしたことを示す。
Proof of sender	id-cti-ets-proofOfSender OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) cti(6) 4 }
	署名者(必ずしも人ではない)が文書を送信したことを示す。(この署名者が文書の生成をしたのかどうかはわからない。)
Proof of approval	id-cti-ets-proofOfApproval OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) cti(6) 5 }
	署名者がその文書を承認したことを示す。
Proof of creation	id-cti-ets-proofOfCreation OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) cti(6) 6 }
	署名者がその文書を生成したことを示す。(この署名者が文書の承認や文書の送信をしたのかどうかはわからない。)

1.1.3.23. Signer location 属性：署名所在地(オプション)

本属性は署名者の所在地情報を格納する属性である。

本属性に格納される居所情報は第三者によってその確かさが保証されたものではなく、署名者が署名時にその場所にいたと主張しているものである。

attrType には次の OID を格納する。

```
ETSI TS 101 733 V1.7.3[18] 5.11.2節 Signer location attributeより
id-aa-ets-signerLocation OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16)
  id-aa(2) 17 }
```

attrValues の値は次に示すデータ構造をしている。

```
ETSI TS 101 733 V1.7.3[18] 5.11.2節 Signer location attributeより
SignerLocation ::= SEQUENCE { -- at least one of the following shall be present
```

```
    countryName [0] DirectoryString OPTIONAL,
        -- As used to name a Country in X.500
    localityName [1] DirectoryString OPTIONAL,
        -- As used to name a locality in X.500
    postalAddress [2] PostalAddress OPTIONAL }
PostalAddress ::= SEQUENCE SIZE(1..6) OF DirectoryString
```

1.1.3.24. Signer attributes 属性：その他の署名者属性情報(オプション)

本属性は本書で定義されていない署名者属性情報を格納したい場合に利用する。

署名者属性情報には大きく分けて

- 署名者が自分でそうだと主張している属性情報(claimed attributes)
- 署名者がそうであると第三者が保証している属性情報(certified attributes)

の２種類があり、本属性にはそのどちらの属性情報も格納することができる。

attrType には次の OID を格納する。

```
ETSI TS 101 733 V1.7.3[18] 5.11.3節 Signer attributes attributeより
id-aa-ets-signerAttr OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16)
 id-aa(2) 18 }
```

attrValues の各値は次に示すデータ構造をしている。

```
ETSI TS 101 733 V1.7.3[18] 5.11.3節 Signer attributes attributeより
SignerAttribute ::= SEQUENCE OF CHOICE{
        claimedAttributes   [0] ClaimedAttributes,
        certifiedAttributes [1] CertifiedAttributes }
ClaimedAttributes ::= SEQUENCE OF Attributes
CertifiedAttributes ::= AttributeCertificate -- as defined in RFC 3281
```

1.1.3.25. Content time-stamp 属性：コンテントタイムスタンプ(オプション)

本属性は署名対象文書に対するタイムスタンプを格納する属性である。

本属性は署名時点より前に取っておいた署名対象文書の存在証明(つまりタイムスタンプ)を署名データに含めたい場合に利用する。さらに署名時刻まで証明したい場合には CAdES-T の章で後述する署名タイムスタンプを用いればよい。

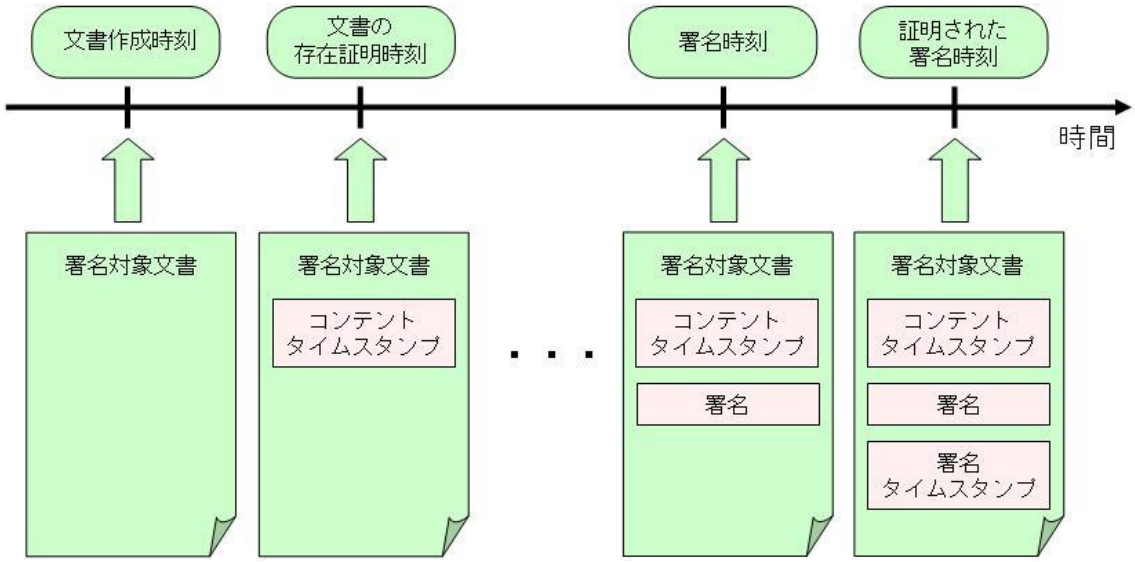


図 1.1.3.25.1: コンテントタイムスタンプが利用されるタイミング

attrType には次の OID を格納する。

```
ETSI TS 101 733 V1.7.3[18] 5.11.4 節 Content time-stamp より
id-aa-ets-contentTimestamp OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16)
 id-aa(2) 20 }
```

attrValues の各値は次に示すデータ構造をしている。

```
ETSI TS 101 733 V1.7.3[18] 5.11.4 節 Content time-stamp より
ContentTimestamp ::= TimeStampToken
```

コンテントタイムスタンプは署名対象文書に対するタイムスタンプなので、コンテントタイムスタンプの TimeStampToken の messageImprint 内に格納するハッシュ値は、必然的に signedData(署名データ)の eContent フィールドの値のハッシュ値(タグ、長さは含めない)に等しくなる。

1.1.3.26. Countersignature 属性：カウンタ署名

本属性は署名値に対する署名データを格納する属性である。

本属性は複数人が同一文書に署名する際、署名の順序に意味がある、もしくは意味を持たせたい場合に用いられる。署名の順序に特に意味がない場合には「付録 2: 複数署名」で述べる並列署名を用いる。(署名順序に意味がある複数署名は直列署名という。)

attrType には次の OID を格納する。

```
RFC 3852[5] 11.4 節 Countersignature より
id-countersignature OBJECT IDENTIFIER ::=
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) 6 }
```

attrValues の各値は次に示すデータ構造とする。

```
RFC 3852[5] 11.4 節 Countersignature より
Countersignature ::= SignerInfo
```

カウンタ署名が CAdES の SignerInfo と異なる点を以下に挙げる。

1．signedAttributes フィールドに Content Type 属性を含めてはならない(MUST NOT[10]11.4)。Message Digest 属性、Signing Certificate 属性は CAdES の SignerInfo と同様に含めなくてはならない(MUST[18]5.7,MUST[10]11.4)。
2．署名対象データはカウンタ署名を付与する署名値(SignerInfo の signature フィールドの値。OCTET STRING のタグ、長さは含まない。)である(MUST[10]11.4)。

本属性は属性値の SignerInfo を複数持つことができ([10]11.4)、独立したカウンタ署名を付与できる。本属性が存在する場合には属性値を必ず 1 つ以上含めなくてはならない(MUST[10]11.4)。

SignerInfo には本属性のインスタンスを複数持たせることができる([10]11.4)。

独立したカウンタ署名を複数付与する場合、以下に示す 2 つの構造は意味的には同じである。

1) 一つの CounterSignatures に複数の SignerInfo 属性値を加える。
2) 複数の CounterSignatures の中に(並列には)一つの SignerInfo 属性のみを持たせる

ECOM では "1)" を推奨する。

カウンタ署名に対するカウンタ署名を付与することも可能であり、そのようにしてカウンタ署名を任意の長さでつなぐこともできる([10]11.4)。

1.2. CAdES-T

CAdES-T（Electronic Signature Time-stamped）は、デジタル署名の存在時刻を確定するために、電子署名文書中の署名値（CMS の SignerInfo の signature フィールド）に対してタイムスタンプ局（TSA）から取得したタイムスタンプトークン（署名タイムスタンプ）

を非署名属性（UnsignedAttributes）に追加したものである。署名値は電子文書のハッシュ値をもとに計算されるため、署名値から生成したタイムスタンプトークンは、署名の存在時刻とともに、電子データの存在時刻も証明することとなる。

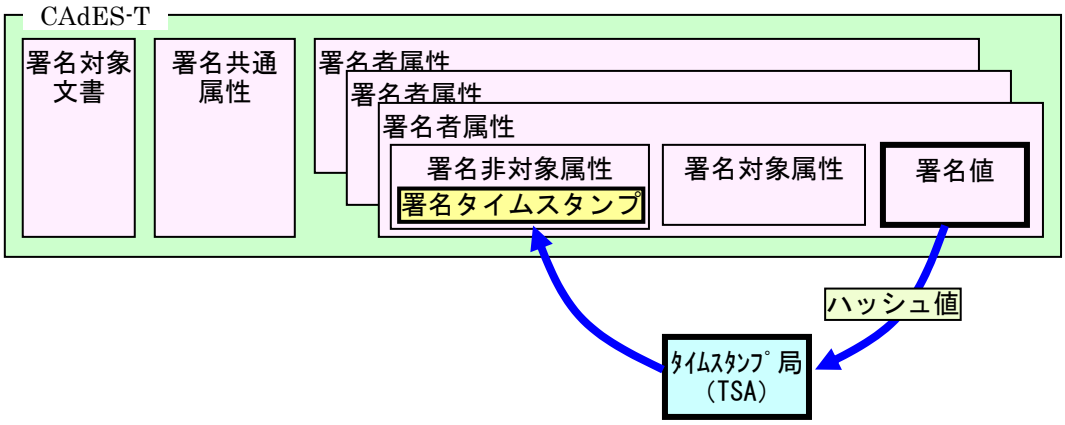


図 1.2.1: 署名タイムスタンプ

ひとつの署名に対していくつかの異なったTSAからタイムスタンプトークンを取得して格納してもよい。複数の署名が添付されている場合、個々の署名値に対してそれぞれタイムスタンプトークンを取得してもよいし、ある署名についてのみタイムスタンプトークンを取得してもよい(MAY[18]6.1.1)。

Signature Timestamp 属性の OID は次の値である。

```
id-aa-signatureTimeStampToken OBJECT IDENTIFIER ::= { iso(1) member-body(2)
                    us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 14 }
```

Signature Timestamp 属性の値は、ASN.1 形式の SignatureTimeStampToken である。

```
SignatureTimeStampToken ::= TimeStampToken
```

TimeStampToken は、RFC 3161([5]2.4.2)で定義されている。

TimeStampToken の messageImprint の値は、CMS の SignerInfo 内の signature フィールド（タイプと長さを除いた値の部分のみ）のハッシュ値である(SHALL[18]6.1.1)。

電子文書の長期保存のために、署名タイムスタンプの検証情報（認証パス及び失効情報）を署名フォーマット内に格納することができる。格納場所については「1.4 CAdES-X」を参照のこと。

1.3. CAdES-C

CAdES-C（Complete validation reference data）は、CAdES-T に対してデジタル署名の検証の際に利用する認証パス上の全ての公開鍵証明書（ただし署名者の公開鍵証明書を

除く）とそれぞれの公開鍵証明書の CRL や OCSP レスポンスなどの失効情報（署名者の公開鍵証明書の失効情報も含む）に対するリファレンス情報を、非署名属性（UnsignedAttributes）に追加したものである。

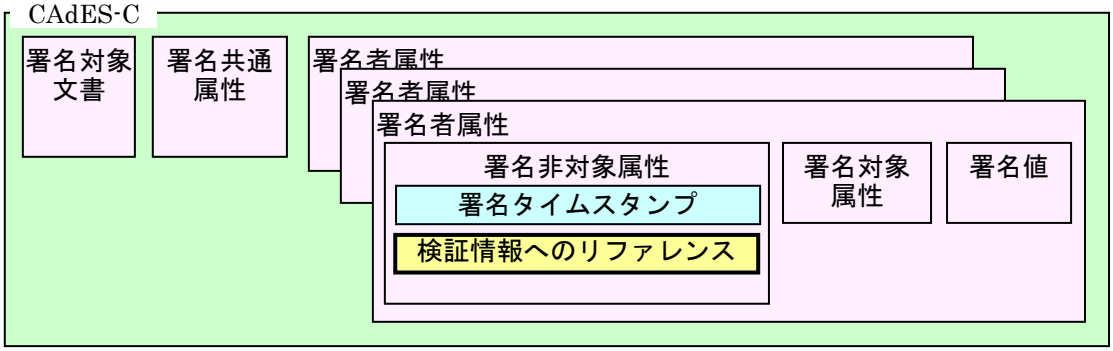


図 1.3.1: CAdES-C

CAdES-C は、署名の検証に必要なすべてのデータ（証明書及び失効情報）に関するリファレンス情報を備える電子署名文書である。

CAdES-C は最小構成として以下を含まねばならない(SHALL[18]6.2)。

- CAdES-T の Signature Timestamp 属性
- Complete Certificate Refs 属性（認証パス上の全ての公開鍵証明書へのリファレンス）
- Complete Revocation Refs 属性（上記各公開鍵証明書の失効情報へのリファレンス）

なお、属性証明書のリファレンス情報を格納する場合の属性として以下が存在するが、JIS[53]では、別途規定を定めなければ使用できないとしている。

- Attribute Certificate Refs
- Attribute Revocation Refs

（1）Complete Certificate Refs 属性の定義

Complete Certificate Refs 属性は unsigned attribute である。Complete Certificate Refs 属性は ES の署名検証に必要な署名者証明書からトラストアンカ証明書までの認証パス上の全ての証明書を参照する。（ただし署名者の証明書への参照は含まない）

この属性を署名データに含める際は、1 つの署名につき一つだけ含めなければならない(SHALL[18]6.2.1)。

注記 1 ：署名者の証明書は signing certificate 属性で参照される。

Complete Certificate Refs 属性の OID は次のとおりである。

```
id-aa-ets-certificateRefs OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                                                  rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 21}
```

Complete Certificate Refs 属性は ASN.1 構文の CompleteCertificateRefs を値として持つ。

```
CompleteCertificateRefs ::= SEQUENCE OF OtherCertID

OtherCertID ::= SEQUENCE {
            otherCertHash            OtherHash,
            issuerSerial             IssuerSerial OPTIONAL
  }

OtherHash ::= CHOICE {
            sha1Hash  OtherHashValue,  -- SHA-1 の場合、ここに格納
            otherHash  OtherHashAlgAndValue
  }

OtherHashValue ::= OCTET STRING

OtherHashAlgAndValue ::= SEQUENCE {
            hashAlgorithm  AlgorithmIdentifier,
            hashValue      OtherHashValue
    }
```

CompleteCertificateRefs に含める OtherCertID の順序については規定がないが、署名者証明書からトラストアンカ証明書の順（但し署名者証明書は含めない）で格納することを ECOM の推奨とする。

OtherCertID の issuerSerial は、ASN.1 構文上では OPTIONAL であるが Complete Certificate Refs を構成する際は必ず含めなければならない。また、otherCertHash に含めるハッシュは参照される証明書のハッシュ値とマッチしなければならない(SHALL[18]6.2.1)。

（2）Complete Revocation Refs 属性の定義

Complete Revocation Refs 属性は unsigned attribute である。この属性を署名データに含める際は、１つの署名に対して一つだけ含めなければならない(SHALL[18]6.2.2)。この属性は、ES の署名検証に必要な署名者からトラストアンカまでの証明書に対する CRL あるいは OCSP レスポンスのすべてを参照する。

Complete Revocation Refs 属性の OID は次のとおりである。

```
id-aa-ets-revocationRefs OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                                        rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 22}
```

Complete Revocation Refs 属性は ASN.1 構文の CompleteRevocationRefs を値として持つ。

```
CompleteRevocationRefs ::= SEQUENCE OF CrlOcspRef

CrlOcspRef ::= SEQUENCE {
            crlids          [0] CRLListID       OPTIONAL,
            ocspids         [1] OcspListID      OPTIONAL,
            otherRev        [2] OtherRevRefs    OPTIONAL
  }
```

CompleteRevocationRefs はシーケンスの先頭に署名者証明書（signing certificate）に対する CrlOcspRef を必ず１つ持たなければならず、続けて CompleteCertificateRefs 属性の中の各 OtherCertID に対して１つずつ CrlOcspRef を持たなければならない。２番目以降の CrlOcspRef の順番は、対応する OtherCertID の順番と同じでなければならない。トラストアンカ証明書を除く認証パス上のすべての証明書に対して、CRLListID、OcspListID、OtherRevRefs のうち、少なくとも一つを含めなければならない(SHALL[18]6.2.2)。

ECOM プロファイルでは、CRL あるいは OCSP レスポンス以外の失効情報（つまり OtherRevRefs）は利用しない([51]1.3)。また JIS[53]では、OtherRevRefs は別途規定を定めなければ使用できないとしている。

```
CRLListID ::=  SEQUENCE {
            crls        SEQUENCE OF CrlValidatedID}

CrlValidatedID ::=  SEQUENCE {
            crlHash                 OtherHash,
            crlIdentifier           CrlIdentifier OPTIONAL}

CrlIdentifier ::= SEQUENCE {
            crlissuer               Name,
            crlIssuedTime           UTCTime,
            crlNumber               INTEGER OPTIONAL
  }
```

```
OcspListID ::=  SEQUENCE {
            ocspResponses        SEQUENCE OF OcspResponsesID}

OcspResponsesID ::=  SEQUENCE {
            ocspIdentifier              OcspIdentifier,
            ocspRepHash                 OtherHash    OPTIONAL
  }

OcspIdentifier ::= SEQUENCE {
            ocspResponderID    ResponderID,
              -- As in OCSP response data
            producedAt      GeneralizedTime
              -- As in OCSP response data
  }
```

CrlValidatedID を作成する際、crlHash は、署名を含む CRL の完全な DER エンコードされたデータに対して計算する。crlIdentifier は、通常、他の情報によって CRL が推測できないときに存在する。

crlIdentifier は、CRL を特定するためのものであり、発行者名と発行時刻を利用している。発行時刻は CRL に含まれる”thisUpdate”が示す時刻でなければならない(SHALL[18]6.2.2)。

CRL が Delta CRL であれば、CRLListID には CRL の集合に対する参照が含まれなければならない(SHALL[18]6.2.2)。

OcspIdentifier は OSCP レスポンスを特定するもので、発行者名と発行時刻を用いる。発行時刻は OCSP レスポンスに含まれる”producedAt”が示す時刻でなければならない(SHALL[18]6.2.2)。同時刻に発行された OCSP を区別するには、OcspResponceID に含まれるレスポンスのハッシュ値を用いる。

```
OtherRevRefs ::= SEQUENCE {
            otherRevRefType      OtherRevRefType
            otherRevRefs         ANY DEFINED BY otherRevRefType
  }

OtherRevRefType ::= OBJECT IDENTIFIER
```

1.4. CAdES-X

CAdES-X（Extended Validation Data）は、将来の CA の危殆化に備えたり、検証データの完全性を確保したり入手困難になることに備えるために、CAdES-C を拡張するものである。

CAdES-X には、CAdES-X Long（図 1.4.1）、CAdES-X Type1（図 1.4.2）、CAdES-X Type2（図 1.4.3）の 3 通りのフォーマットが用意される。ECOM プロファイルでは、CAdES-X Long の使用のみを認めている([51]1.4)。

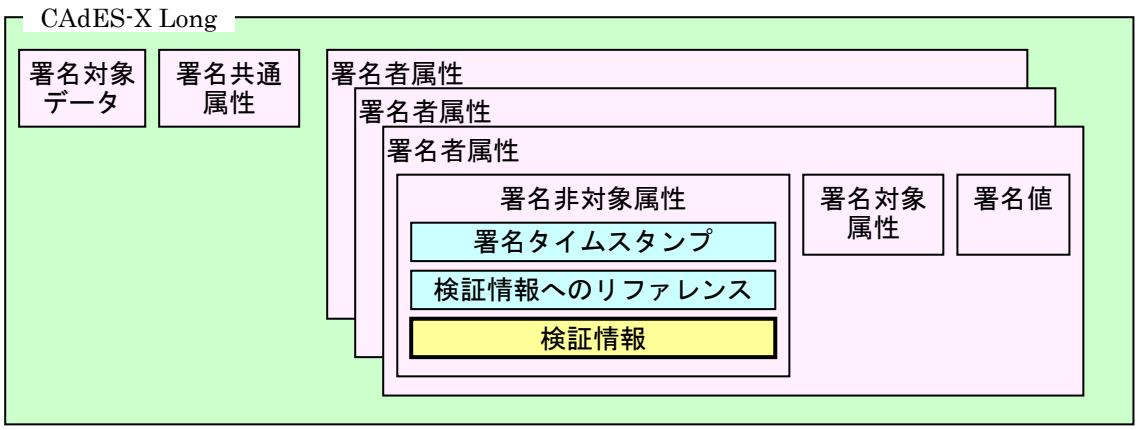


図 1.4.1: CAdES-X Long

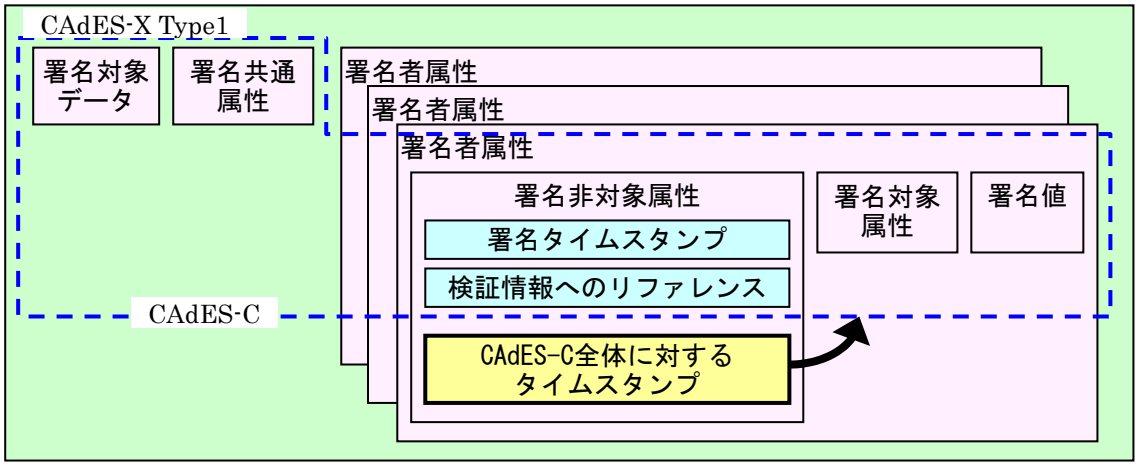


図 1.4.2: CAdES-X Type1

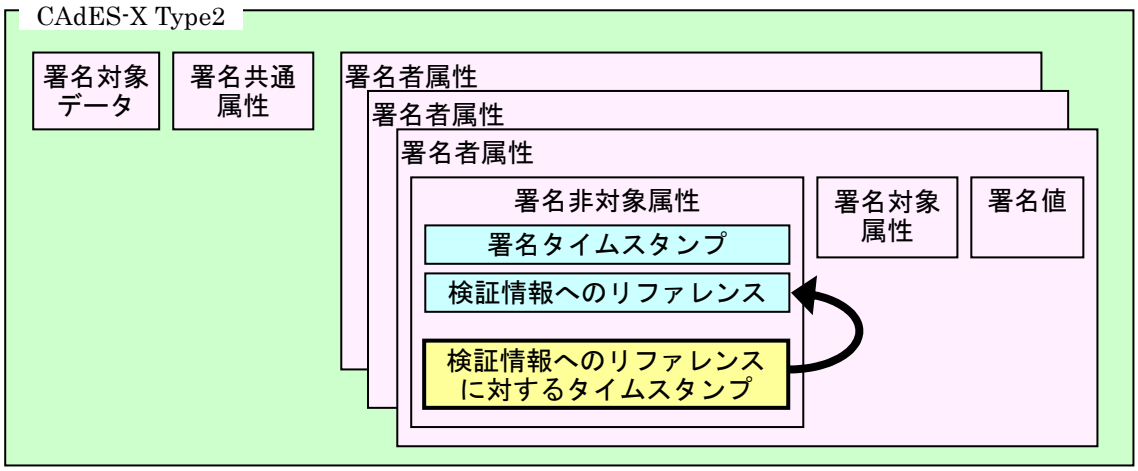


図 1.4.3: CAdES-X Type2

CAdES-X Long は検証情報そのものを署名データ内に抱え込むフォーマットである。CAdES-X Type1 は、CAdES-C 全体に対するタイムスタンプを取得して追加するもの、CAdES-X Type2 は、検証情報へのリファレンスのみに対するタイムスタンプを取得して追加するものである。

検証情報のリファレンスには検証情報のハッシュ値が含まれるが、そのとき用いるハッシュ関数が脆弱化するケースを想定すると、リファレンスをタイムスタンプの対象とするのでは、リファレンスと検証情報そのものとの対応関係を証明することができなくなる。更に、検証情報そのもの（特に中間のサブ CA の公開鍵証明書や失効情報など）の消失に備えるには、検証情報そのものを保持しておく必要がある。

長期保存のためには、署名対象データ、デジタル署名、タイムスタンプ、検証情報全体をタイムスタンプや耐タンパな仕組みで保護する必要がある。長期署名フォーマットでは、この後に述べるアーカイブタイムスタンプによって保護する。つまり、適切な時期にアーカイブタイムスタンプを追加することによって、CAdES -C タイムスタンプも検証情報リファレンスへのタイムスタンプも必要なくなり、重要なのは検証情報そのものを確保しておくことである。

CAdES-X Long は保護対象となる全てのデータを格納するフォーマットに相当する。電子署名文書の長期保存を可能とするシステムを実装するためには、CAdES-X Long のみをサポートすればよい。

以下で、CAdES-X を構成するための属性について説明する。

（1）Certificate Values 属性の定義

この属性は、CompleteCertificationRefs が参照する証明書および署名者の証明書を保持するもので、CAdES-X Long で使用される。Certificate Values 属性は unsigned attribute である。この属性は 1 署名につき 1 つだけ存在する。

注 1：CAdES v1.4.0 では、署名者証明書の格納場所は指定されておらず、また、SignedData の certificates ではアーカイブタイムスタンプの対象とならず保護されないため、ここに署名者証明書を含める必要がある。

注 2：Certificate Values 属性には属性証明書を含めることはできない。現行の CAdES v1.7.3 では、CMS の属性に属性証明書そのものを格納できる属性がない。XAdES では属性証明書を格納するプロパティが新設されたため、CAdES についても含めることができるよう ECOM から ETSI TC ESI に対して働きかけを行っていく予定である。現時点では、属性証明書を certificates フィールドに格納するか、属性証明書の持つ資格情報などの属性情報を証明書の形ではなく SignerInfo 中の signer-attributes 属性として持たせる必要がある。

Certificate Values 属性の OID は次のとおりである。

```
id-aa-ets-certValues OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                                rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 23}
```

Certificate Values 属性は値として次の ASN.1 構文で表される CertificateValues を取る。

```
CertificateValues ::=  SEQUENCE OF Certificate
```

Certificate の定義は RFC 3280([6]4.1)と ITU-T Recommendation X.509([24]A.1)を参照のこと。

（2）Revocation Values 属性の定義

この属性は、Complete Revocation Refs 属性で参照される CRL と BasicOCSPResponse の値を保持するもので、CAdES-X Long で使用される。Revocation Values 属性は unsigned attribute である。この属性は 1 署名につき 1 つだけ存在する。

Revocation Values 属性の OID は次のとおりである。

```
id-aa-ets-revocationValues OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                                rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 24}
```

Revocation Values 属性は値として次の ASN.1 構文で表される RevocationValues を取る。

```
RevocationValues ::=  SEQUENCE {
    crlVals           [0] SEQUENCE OF CertificateList     OPTIONAL,
    ocspVals          [1] SEQUENCE OF BasicOCSPResponse   OPTIONAL,
    otherRevVals      [2] OtherRevVals
}

OtherRevVals ::= SEQUENCE {
    otherRevValType       OtherRevValType,
    otherRevVals          ANY DEFINED BY otherRevValType
}

OtherRevValType ::= OBJECT IDENTIFIER
```

※Other Revocation Values は、ECOM プロファイルでは利用していない([51]1.4)。

※CertificateList の定義は、RFC 3280([6]5.1)と ITU-T Recommendation X.509([24]A.1)を参照のこと。

※BasicOCSPResponse の定義は、RFC 2560([1]4.2.1)を参照のこと。

（3）CAdES-C Time-Stamp 属性の定義

この属性は、CAdES-C 全体に対するタイムスタンプトークンを保持するもので、CAdES-X Type1 で使用される。CAdES-C Time-Stamp 属性は unsigned attribute である。この属性は 1 署名につき複数存在できる。

※JIS[53]では、別途規定を定めなければ使用できないとしており、ECOM プロファイルではこの属性は利用していない([51]1.4)。

CAdES-C Time-Stamp 属性の OID は次のとおりである。

```
id-aa-ets-escTimeStamp OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                                rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 25 }
```

CAdES-C Time-Stamp 属性は、値として次の ASN.1 構文で表される ESCTimeStampToken を取る。

```
ESCTimeStampToken ::=  TimeStampToken
```

（4）Time-Stamped Certificates and CRLs References 属性の定義

この属性は、検証情報へのリファレンス（Complete Certification Refs 属性および Complete Revocation Refs 属性）に対するタイムスタンプトークンを保持するもので、CAdES-X Type2 で使用される。Time-Stamped Certificates and CRLs References 属性は unsigned attribute である。

※JIS[53]では、別途規定を定めなければ使用できないとしており、ECOM プロファイルではこの属性は利用していない([51]1.4)。

Time-Stamped Certificates and CRLs References 属性の OID は次のとおりである。

```
id-aa-ets-certCRLTimestamp OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                                    rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 26}
```

Time-Stamped Certificates and CRLs References 属性は、値として次の ASN.1 構文で表される TimestampedCertsCRLs を取る。

```
TimestampedCertsCRLs ::=  TimeStampToken
```

補足：Complete Validation Reference および Extended Validation Data に含めるデータ

- CompleteCertificateRefs の署名者証明書のリファレンス情報は、Signing Certificate 属性にあるため、CompleteCertificateRefs には含めない。
- トラストアンカの失効情報は通常存在しないため、CompleteRevocationRefs および RevocationValues 中にはトラストアンカのデータは格納しない。トラストアンカの失効情報が存在する場合には、含めてもよい。
- CompleteRevocationRefs の先頭には署名者証明書の失効情報リファレンスを含めるものとする(SHALL[18]6.2.2)。また、CompleteCertificateRefs と CompleteRevocationRefs の各要素の順序は同じ順序であるものとする(SHALL[18]6.2.2)。これ以外に格納順序に関する規定は無いが、以下の属性において署名者からトラストアンカの順で要素を格納することを ECOM の推奨とする。但し、検証時は相互運用性の観点からこの順序を前提として処理すべきではない。

  CompleteCertificateRefs
  CompleteRevocationRefs
  CertificateValues
  RevocationValues

ETSI TS 101 733 v1.7.3[18]の規定による格納方法を以下に示す。

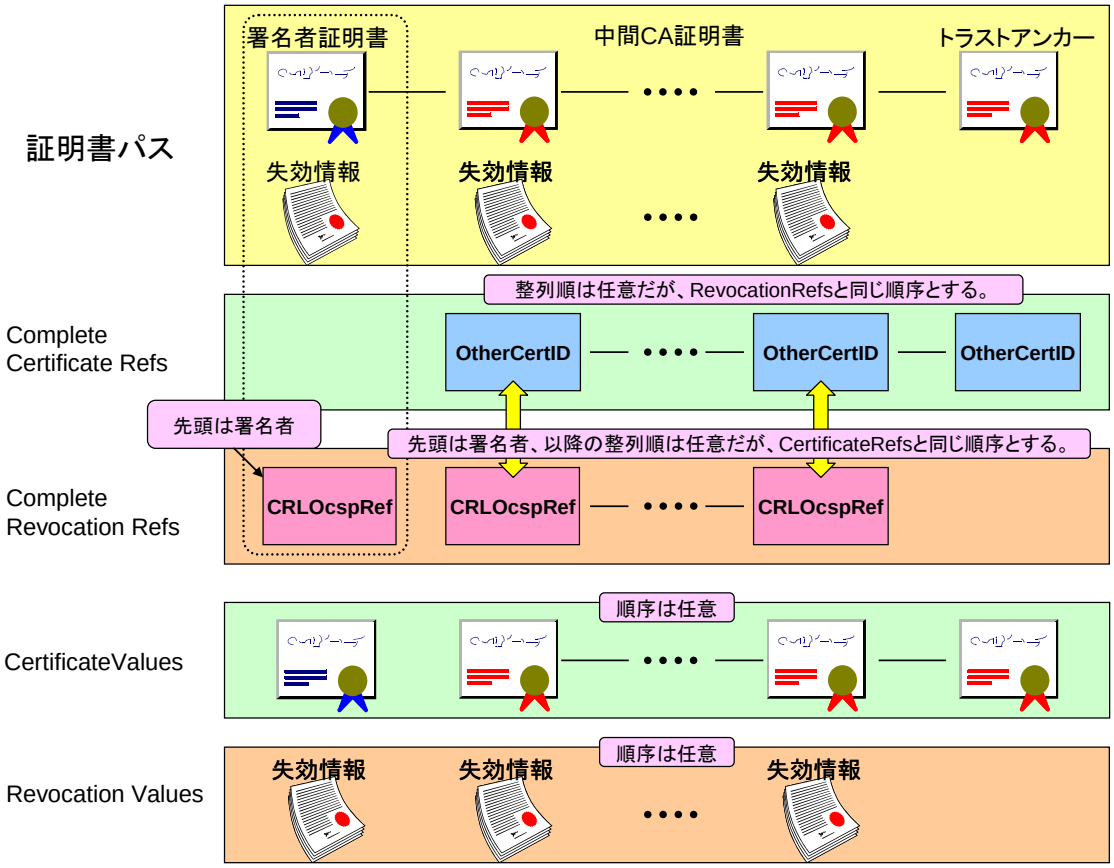


図 1.4.4 ETSI TS 101 733 で規定された制限の緩い検証情報格納方法

次に ECOM が推奨する検証情報の格納方法を示す。

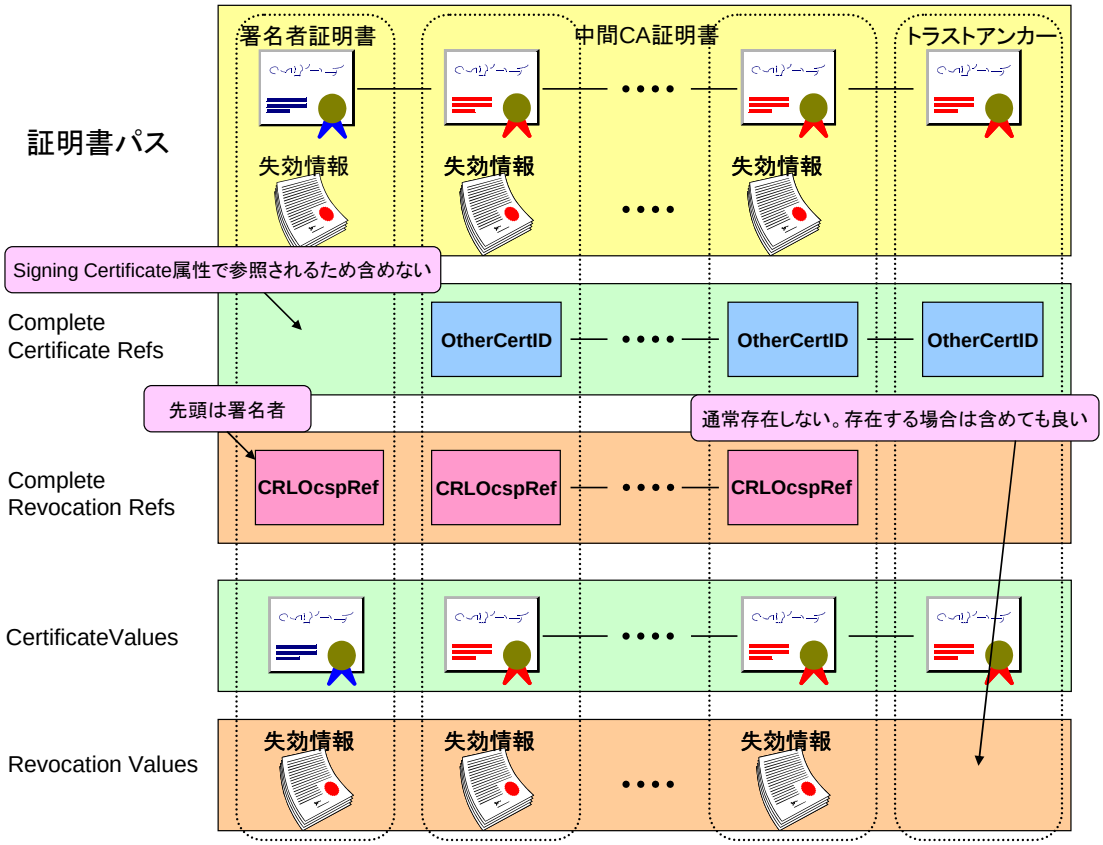


図 1.4.5: ECOM で推奨する検証情報の格納内容および順序

以上で署名者の署名に対する検証情報の格納について説明した。

一方、CAdES-T の署名タイムスタンプの検証情報（認証パス及び失効情報）については、以下の格納場所がある。（後述する CAdES-A のアーカイブタイムスタンプの検証情報についても同様である。）

① タイムスタンプトークン内の certificates および crls フィールド

② タイムスタンプトークン内の非署名属性群に入れる

本書では、構築時には②に格納することを推奨し、検証時には①,②を処理できることを必須とする。

タイムスタンプトークン内の非署名属性に CAdES-C, CAdES-X Long で規定された属性により検証情報を含めることができる(②)(MAY[18]6.2, 6.3)。これらの属性を使う場合にはタイムスタンプトークン内の非署名属性群(Complete Validation Reference Data と Extended Validation Data 形式)に格納するものとする(SHALL[18]6.2.1, 6.2.2, 6.3.3, 6.3.4)。なお、署名タイムスタンプの検証情報が CA や TSA などの信頼される第三者機関(TTP)において安全に保管されており、それを必要に応じて参照できる場合は、格納しないことを選択することができる。

※Complete Validation Reference Data、Extended Validation Data についてはそれぞれ「1.3. CAdES-C」、「1.4. CAdES-X」を参照。検証情報の格納形式は署名者の証明書の場合と同じであり、「署名者の証明書」を「タイムスタンプ局の証明書」へ読み替えることで適用できる。

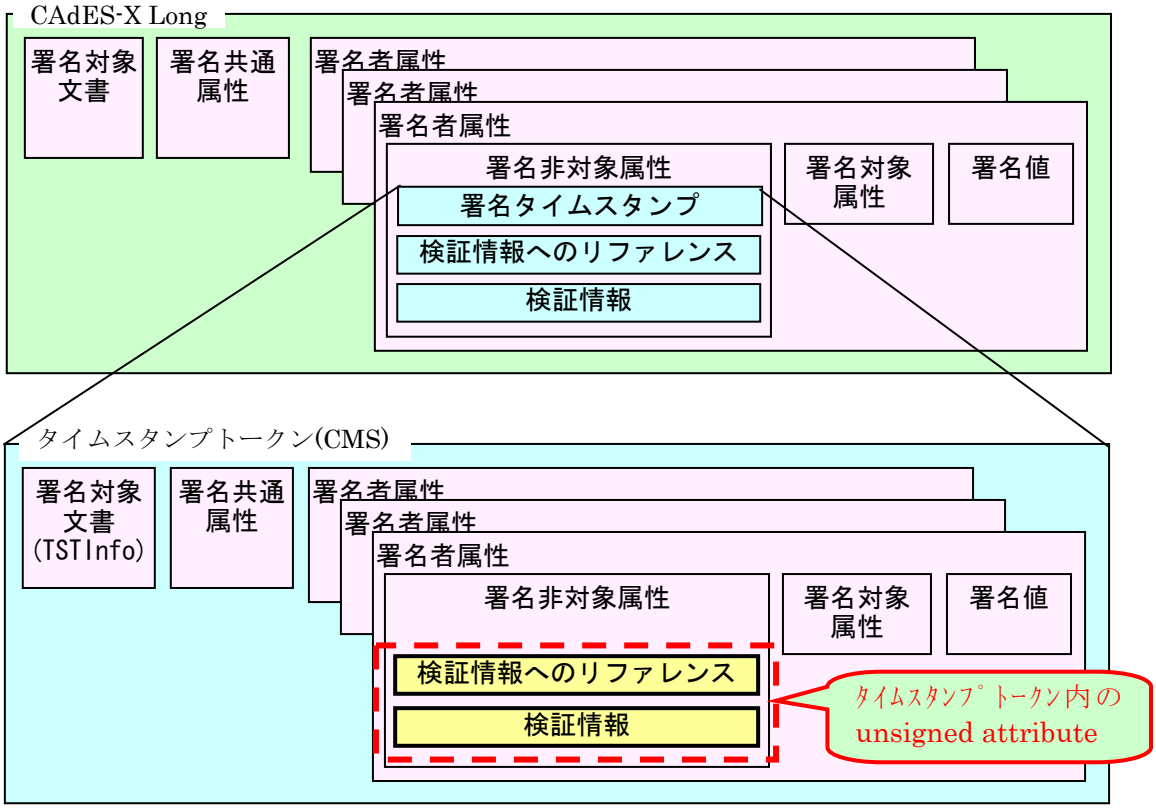


図 1.4.6: 署名タイムスタンプの検証情報の格納場所

1.5. CAdES-A

電子署名の検証可能期間を極めて長くしようとしたとき、証明書、署名、タイムスタンプなどで使用される暗号アルゴリズムの危殆化や TSA の証明書の有効期限切れが発生しうる。この対策として、検証情報を保存し(CAdES-X Long 構築)、アーカイブタイムスタンプを付与する(CAdES-A 構築)ことで、電子署名の検証可能期間をアーカイブタイムスタンプの有効期限まで延長することができる。アーカイブタイムスタンプを複数回重ねることにより、さらに電子署名の検証可能期間を延長することができる。アーカイブタイムスタンプは、署名対象データと署名全体（電子署名やタイムスタンプの検証情報も含む）に対するタイムスタンプであり、Archive Time-Stamp 属性が用いられる。Certificate Values と Revocation Values 属性が無い場合、アーカイブタイムスタンプを付与する前にこれらの属

性を追加しなければならない(SHALL[18]6.4.1)。Archive Time-Stamp 属性は、unsigned attribute である。この属性は、1 署名に対して、時間の経過や複数の TSA から得ることにより複数添付できる。タイムスタンプは、オリジナルの署名よりも強いアルゴリズム（あるいは長い鍵）を利用するのが望ましい。

TimeStampToken に関しては、RFC 3161([5])を参照のこと。

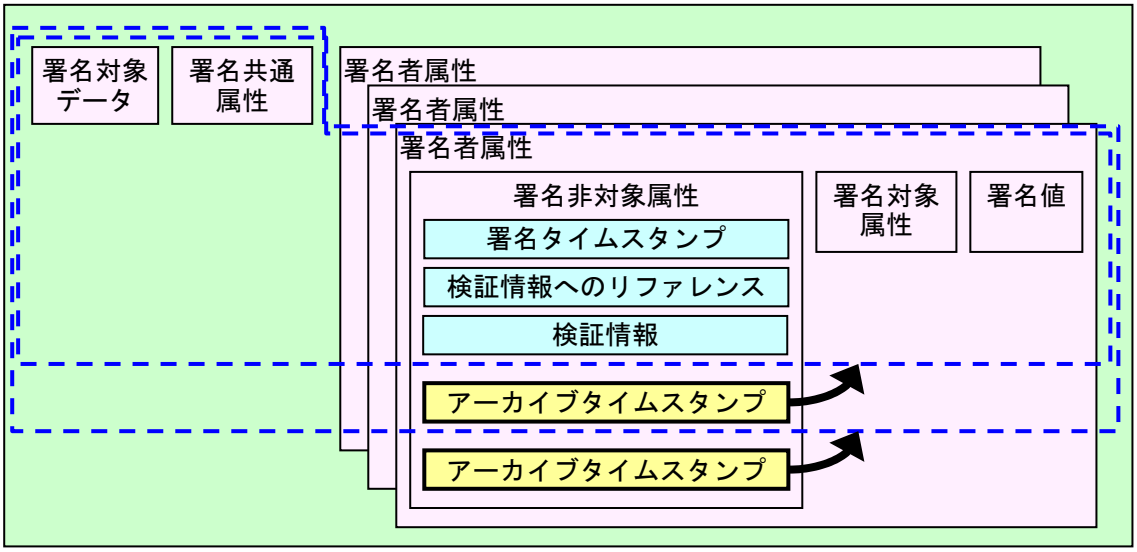


図 1.5.1: CAdES-A

CAdES の仕様である ETSI TS 101 733 では、アーカイブタイムスタンプの対象とするハッシュ値の生成方式が、以下のバージョンにより異なる。

・ETSI TS 101 733 V1.4.0 以前
・ETSI TS 101 733 V1.5.1 ～ V1.6.3
・ETSI TS 101 733 V1.7.3

本書では、新たに実装するものであれば、国内利用、国際利用を問わず V1.7.3 ベースとすることを推奨する。また、後方互換性を必要とするシステムであれば、2005 年 ECOM プロファイルに基づく V1.5.1, V1.4.0 ベースの実装とすることを推奨する。

V1.5.1 および V1.6.3 は CounterSignature なども踏まえた汎用的な仕様とするため、V1.4.0 から大幅に変更を加えたものであるが、不確定要素が多く相互運用やデータ交換を行う場合には、署名者や検証者側で取り決めが必要となる。これらの問題点について ETSI TC ESI と ECOM とで 1 年以上かけて提言、調整を行い相互運用上の問題点を解決した仕様が V1.7.3 である。CAdES プロファイルの JIS 標準も V1.7.3 に基づいている。V1.5.1～V1.6.3 のアーカイブタイムスタンプ仕様には準拠すべきではない。

以下、それぞれの方式について説明する。

1.5.1. ETSI TS 101 733 V1.4.0 以前

（1）V1.4.0 以前の Archive Time-Stamp 属性の定義

Archive Time-Stamp 属性の OID は次のとおりである。

```
id-aa-ets-archiveTimestamp OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                                    rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 27}
```

Archive Time-Stamp 属性は、値として次の ASN.1 構文で表される ArchiveTimeStampToken を取る。

```
ArchiveTimeStampToken ::=  TimeStampToken
```

（2）V1.4.0 以前のアーカイブタイムスタンプのハッシュ値生成対象・方法

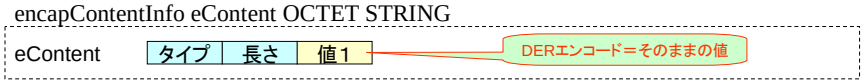
TimeStampToken の messageInprint の値は、次のデータをこの順に結合した値（ただし、タイプと長さを除いたもの）のハッシュ値である。

- encapContentInfo eContent OCTET STRING
- signedAttributes
- signature field within SignerInfo
- SignatureTimeStampToken attribute
- CompleteCertificateRefs attribute
- CompleteRevocationRefs attribute
- CertificateValues attribute（まだこの値を確保していなければ、CAdES-A を作る際に確保しなければならない。）
- RevocationValues attribute（まだこの値を確保していなければ、CAdES-A を作る際に確保しなければならない。）
- 既に追加されている ArchiveTimeStampToken attributes（古い順に並べる）

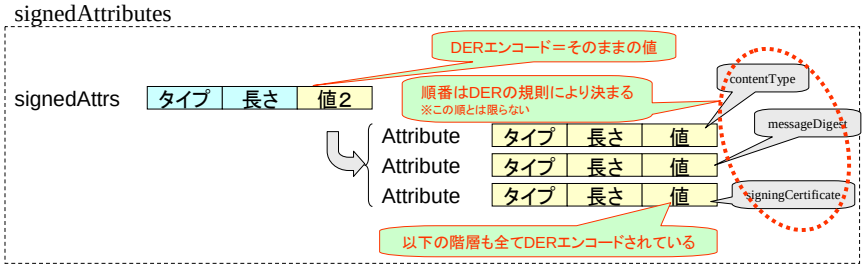
ちなみに、以下の属性は ETSI TS 101 733 V1.4.0 ではハッシュ対象とされているが、ECOM プロファイルでは利用していない（対象に含めない）([51]1.5)。

- CAdES-C TimeStampToken attribute
- TimestampedCertsCRLs attribute

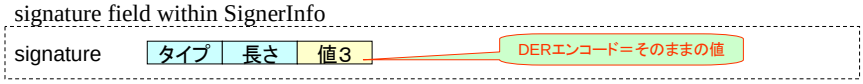
encapContentInfo eContent OCTET STRING



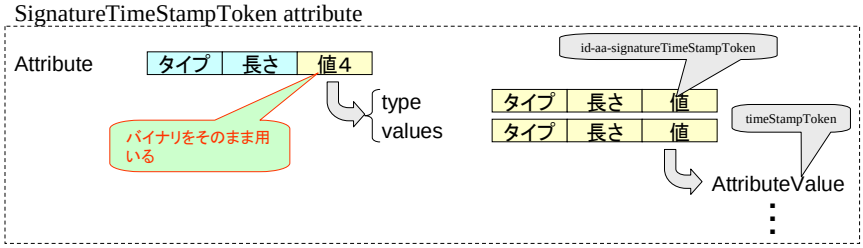
signedAttributes



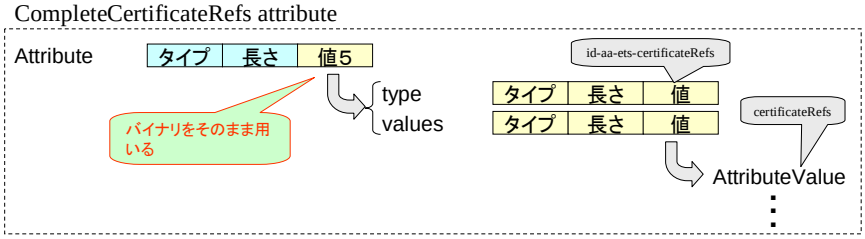
signature field within SignerInfo



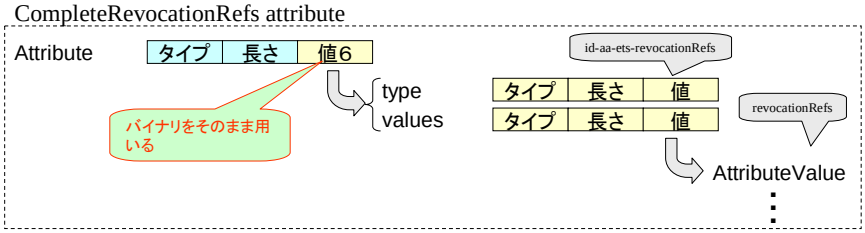
SignatureTimeStampToken attribute



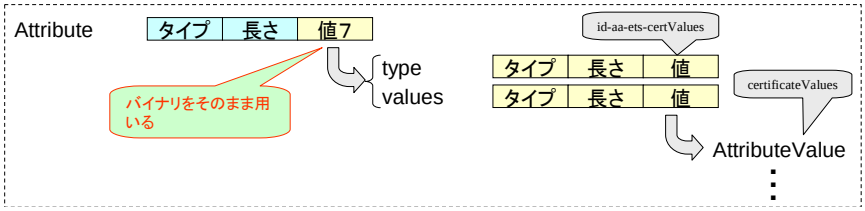
CompleteCertificateRefs attribute



CompleteRevocationRefs attribute



CertificateValues attribute



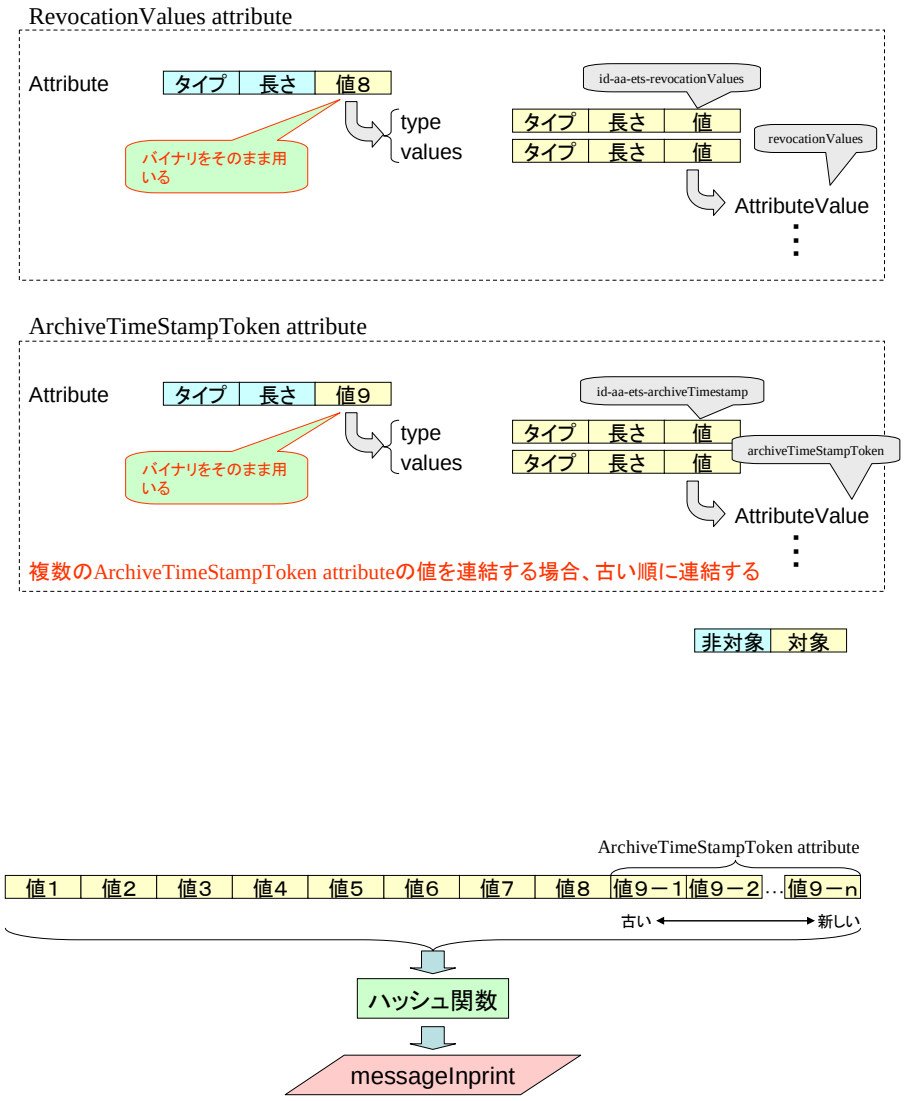


図 1.5.2 V1.4.0 以前でのハッシュ生成方法

1.5.2. ETSI TS 101 733 V1.5.1 ～ V1.6.3

（1）V1.5.1 ～ V1.6.3 の Archive Time-Stamp 属性の定義

Archive Time-Stamp 属性の OID は次のとおりである。V1.4.0 以前と同じである。

```
id-aa-ets-archiveTimestamp OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                                                rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 27 }
```

Archive Time-Stamp 属性は、値として次の ASN.1 構文で表される ArchiveTimeStampToken を取る。

```
ArchiveTimeStampToken ::=  TimeStampToken
```

（2）V1.5.1 ～ V1.6.3 のアーカイブタイムスタンプのハッシュ値生成対象・方法

TimeStampToken の messageInprint の値は、次のデータをこの順に結合した値のハッシュ値である。（タイプと長さを除かない）

- SignedData シーケンスの encapContentInfo（注 1）
- SignedData シーケンスの Certificates と crls（存在する場合）
- SignerInfo シーケンス内のすべての要素

注１：外部署名の場合、V1.5.1～V1.6.3 では、署名対象データはアーカイブタイムスタンプのハッシュ対象とはならない。encapContentInfo は外部署名であるかどうかにかかわらず、そのままをハッシュ対象とする。このため、外部署名の場合、署名対象データは直接的にはアーカイブタイムタイムスタンプの保護対象となっていない。

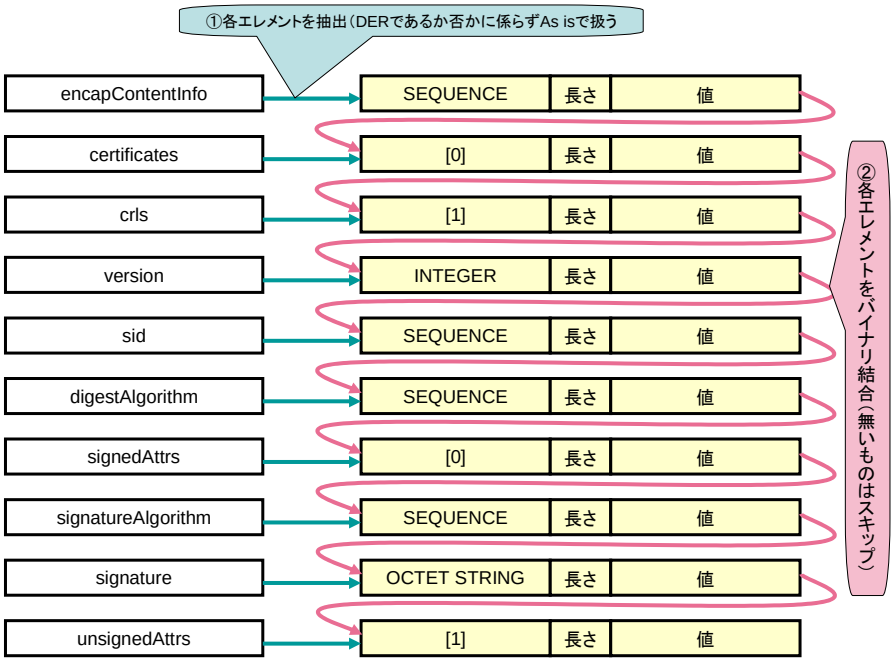


図 1.5.3 V1.5.1～V1.6.3 でのハッシュ生成方法

1.5.3. ETSI TS 101 733 V1.7.3

（1）V1.7.3 の Archive Time-Stamp 属性の定義

V1.7.3 の Archive Time-Stamp 属性の OID は次のとおりである。ArchiveTimeStampV2 属性であるとし、V1.6.3 以前とは別の OID が割り当てられている。

```
id-aa-ets-archiveTimestamp OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                                                rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 48 }
```

Archive Time-Stamp 属性は、値として次の ASN.1 構文で表される ArchiveTimeStampToken を取る。

```
ArchiveTimeStampToken ::=  TimeStampToken
```

（2）V1.7.3 のアーカイブタイムスタンプのハッシュ値生成対象・方法

TimeStampToken の messageInprint の値は、次のデータをこの順に結合した値のハッシュ値である。（タイプと長さを除かない）

- SignedData シーケンスの encapContentInfo
- encapContentInfo の eContent が省略されている場合、署名対象である外部のコンテント
- SignedData シーケンスの certificates と crls（存在する場合）
- SignerInfo シーケンス内のすべての要素

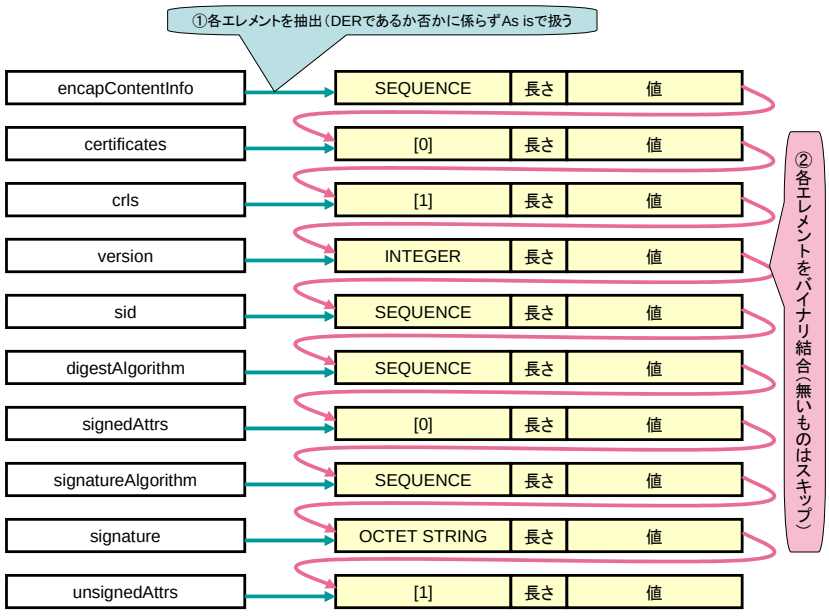


図 1.5.4 V1.7.3 でのハッシュ生成方法（eContent が省略されていない場合）

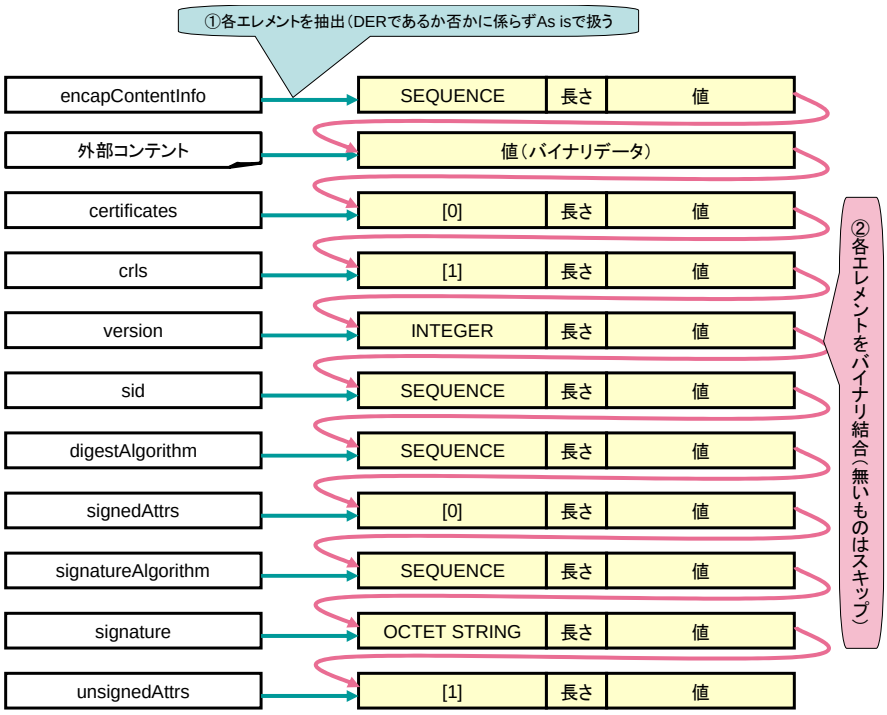


図 1.5.5 V1.7.3 でのハッシュ生成方法（eContent が省略されている場合）

＜注意事項＞

- 生成の際には、signedAttributes の ASN.1 TLV 構造のバイト列を変更してはならないのはもちろんであるが、unsignedAttributes の各要素について、順序ならびに各属性を構成するバイト列を変更してはならない。
- CAdES-A の生成、もしくは、署名延長時にアーカイブタイムスタンプを追加する場合には、unsignedAttributes の[1]IMPLICIT SET OF 構造の最後に属性を追加する。
- 各世代のアーカイブタイムスタンプを検証するために、アーカイブタイムスタンプのハッシュ値を計算する際、検証対象のアーカイブタイムスタンプに応じた unsignedAttributes の再構築が必要となるが、その際には以下のことに注意する。
  - 時間的に検証対象アーアイブタイムスタンプ以降のアーカイブタイムスタンプ属性のみを除いて SET OF 構造を再構築する。
  - 非署名属性の順序を入れ替えてはならない。当然、ソートもしてはならない。
  - 各非署名属性を構成する ASN.1 構造のバイト列の内容を一切変更してはならない。
- カウンタ署名に対してアーカイブタイムスタンプを取得する場合は、カウンタ署名単体で CAdES-A を構築するのではなく、カウンタ署名が含まれる SignedData 構造に対して取得する。また、アーカイブタイムスタンプ取得以降

は、アーカイブタイムスタンプ以外の属性（カウンタ署名含む）を追加してはならない。

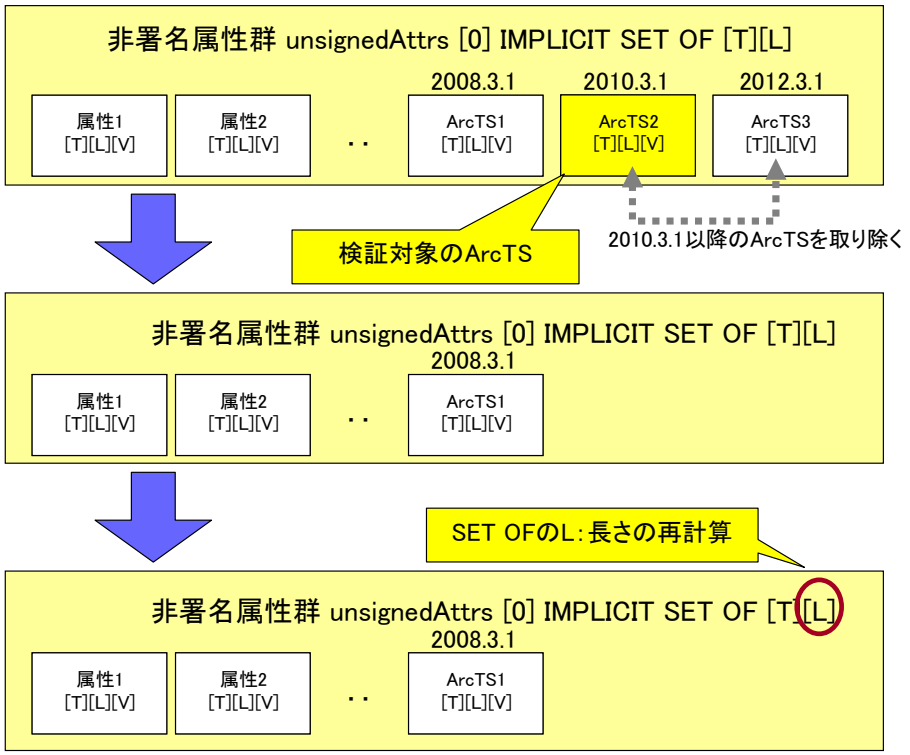


図 1.5.6 V1.7.3 アーカイブタイムスタンプ検証での非署名属性の再構築

1.5.4. アーカイブタイムスタンプの検証情報の格納場所

アーカイブタイムスタンプの検証情報は次のいずれかに格納することが考えられる。

① タイムスタンプトークン内の certificates および crls フィールド

② タイムスタンプトークン内の非署名属性群(Complete Validation Reference Data と Extended Validation Data 形式)に入れる

上記格納場所および形式については、「1.4. CAdES-X」で説明しているので、参照されたい。

2. XAdES

XAdES(XML Advanced Electronic Signatures) は 、 XML 署 名 (W3C/IETF Recommendation (February 2002):"XML-Signature Syntax and Processing") [32]をベースとした長期署名フォーマットで、ETSI により策定されている。現在までに、W3C の Note[31]としても公開されている ETSI TS 101 903 V1.1.1 (2002-02) [19]、ETSI TS 101 903 V1.2.2(2004-4) [20]、および最新版である ETSI TS 101 903 V1.3.2(2006-03) [21]の複数のバージョンが存在する。本稿では、現在の最新版である V1.3.2(2006-03)に準拠する形で示す。本章では、はじめにベースとなる XML 署名について説明し、次に XAdES について詳説する。

2.1. XML 署名

2.1.1. XML 署名の特徴

XML 署名は XML で記述されたデジタル署名フォーマットであり、前述の通り XML 署名にて規定されている。

XML 署名の特徴として、

- XML 署名自身がテキスト表現の XML データであるため CMS に比べ可読性が良い。
- 他の XML データとの親和性が高く署名対象となる XML 文書中に XML 署名を格納することや、XML 署名中に署名対象の XML データを格納することが可能である。
- 署名対象への参照を URI で表現しており、この URI の持つ強力な参照機能により外部データへの参照や XML 文書の各要素への参照が可能である。
- 部分署名や多重署名といった複雑な署名が可能である。

などがあげられる。

2.1.2. XML 署名の形式

XML 署名はその署名対象の位置により以下の 3 種類に分類される。

1)Enveloped 形式

署名対象が XML 署名(Signature 要素)に対し祖先要素である場合を Enveloped 形式の XML 署名という。なお実際に署名を生成または検証する場合には、その XML 署名自身が署名対象として含まれないように計算が行われる。

2)Enveloping 形式

署名対象が XML 署名中に包含されている場合を Enveloping 形式の XML 署名という。XML 署名の規格上、署名対象は後述する Object 要素に包含される。

3)Detached 形式

署名対象の要素や文書が XML 署名(Signature 要素)と親子関係の無い場合を Detached 形式の XML 署名という。署名対象は XML 署名文書とは同一ファイルでも別ファイルでもよく、また別ファイルの場合任意のフォーマットの文書でもよい。

これら 3 つの形式の概要を図 2.1.2.1 に示す。

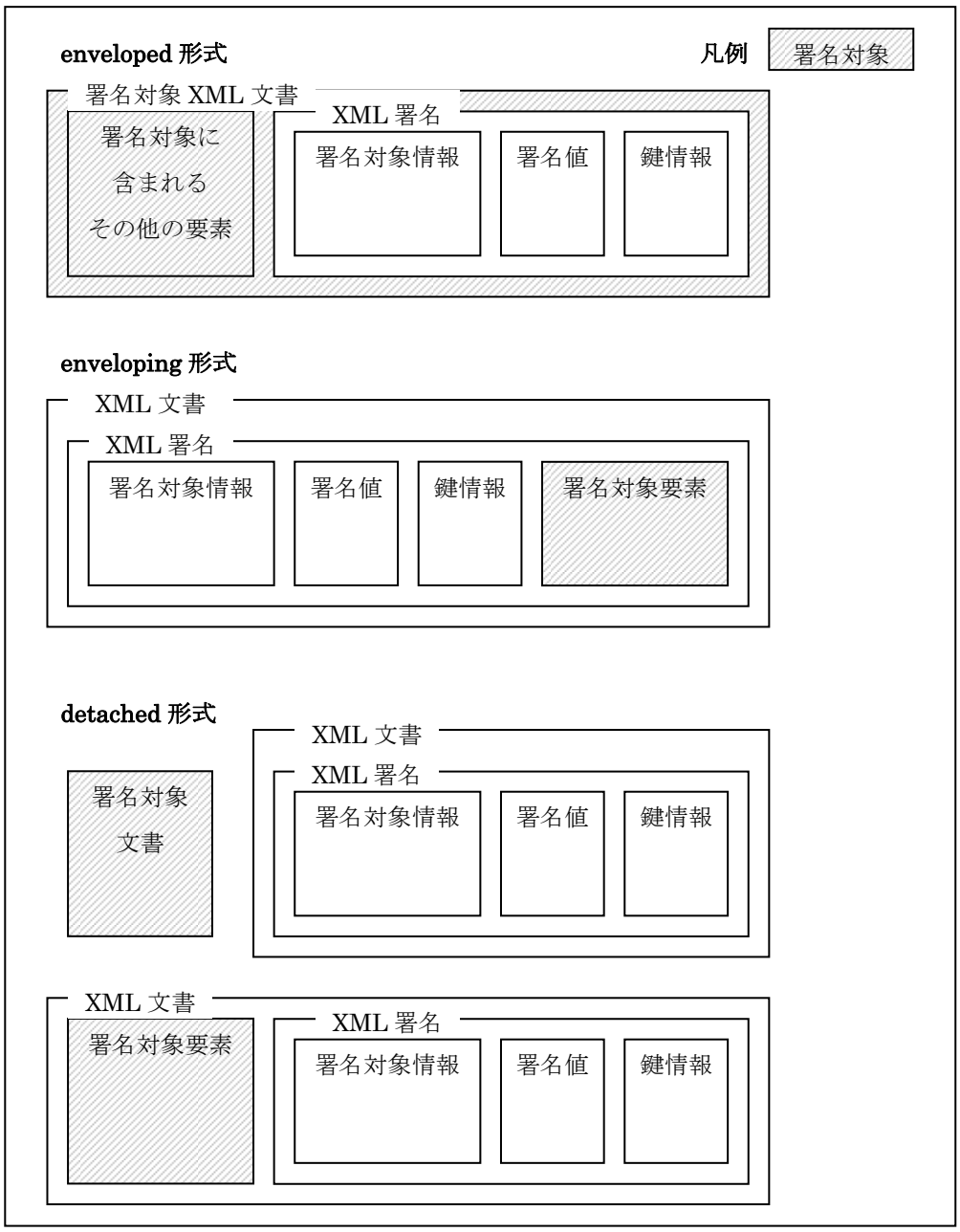


図 2.1.2.1: XML 署名の 3 つの形式

2.1.3. XML 署名のフォーマット

XML 署名の基本フォーマットの概要をリスト 2.1.3.1 に示す。

リスト 2.1.3.1: XML 署名の基本フォーマットの概要

```
<Signature ID?>
     <SignedInfo>
       <CanonicalizationMethod/>
       <SignatureMethod/>
       (<Reference URI? >
         (<Transforms>)?
         <DigestMethod>
         <DigestValue>
       </Reference>)+
     </SignedInfo>
     <SignatureValue>
    (<KeyInfo>)?
    (<Object ID?>)*
</Signature>
```

※ ここで"?"は 0 または 1、"+"は 1 以上、"*"は 0 以上存在することを意味する。

XML 署名の計算は主に署名対象を参照しダイジェスト値を計算する処理と署名対象情報要素から署名値を計算する処理から構成される。図 2.1.3.1 に XML 署名での全体の計算の流れを、図 2.1.3.2 に署名対象の参照からダイジェスト値計算までの流れを、図 2.1.3.3 に署名対象情報要素から署名値計算の流れを示す。

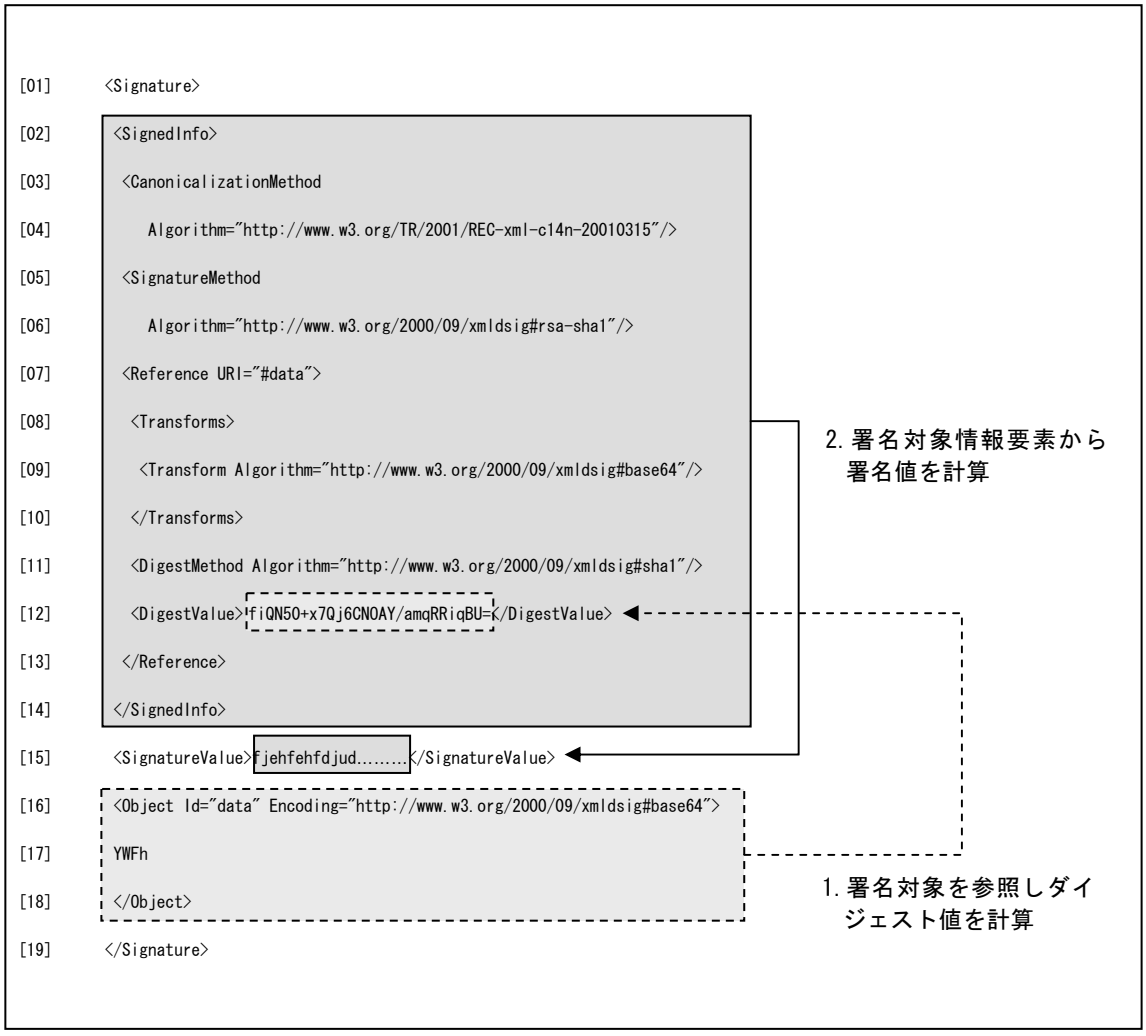


図 2.1.3.1: XML 署名の全体の計算の流れ

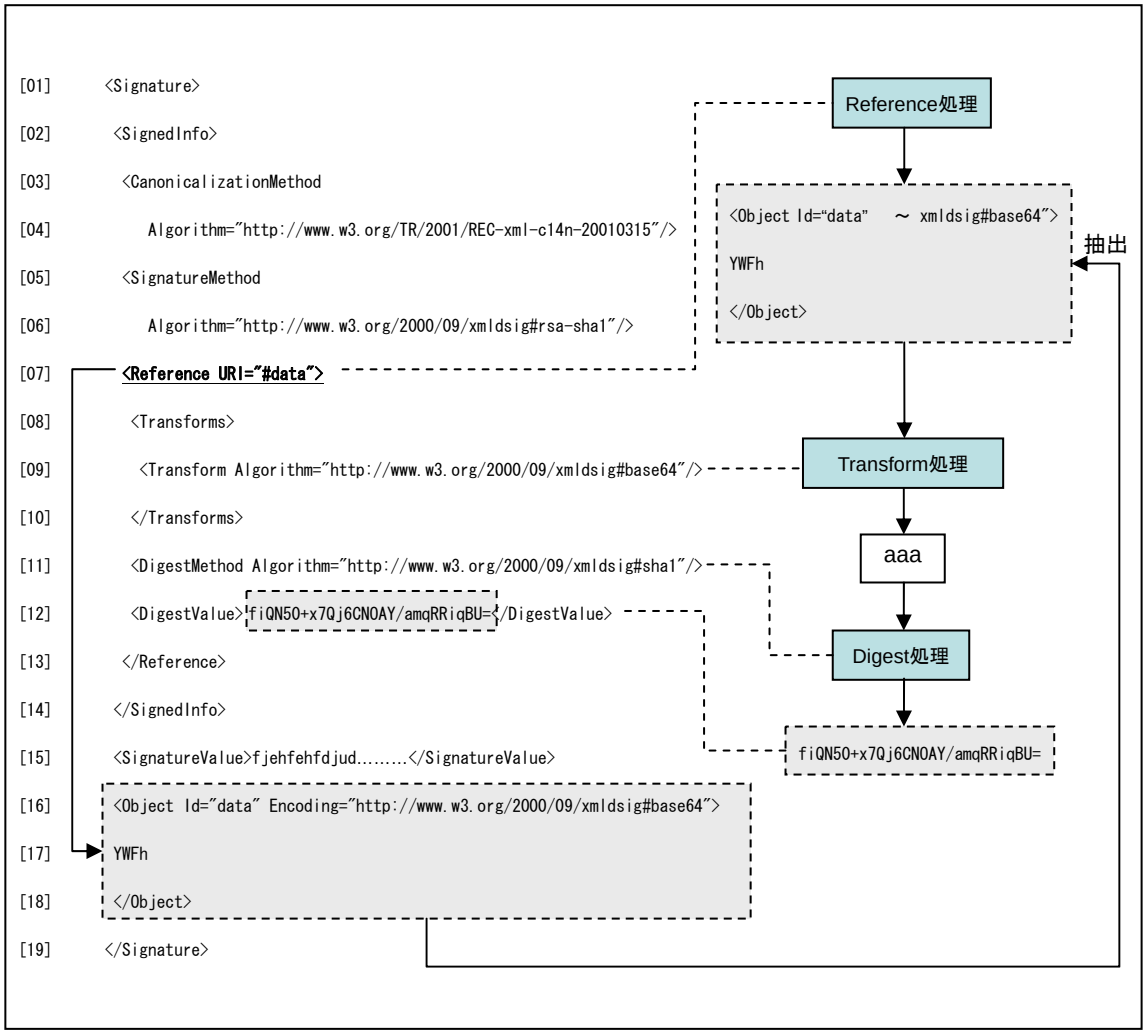


図 2.1.3.2: Reference 処理から Digest 計算までの流れ

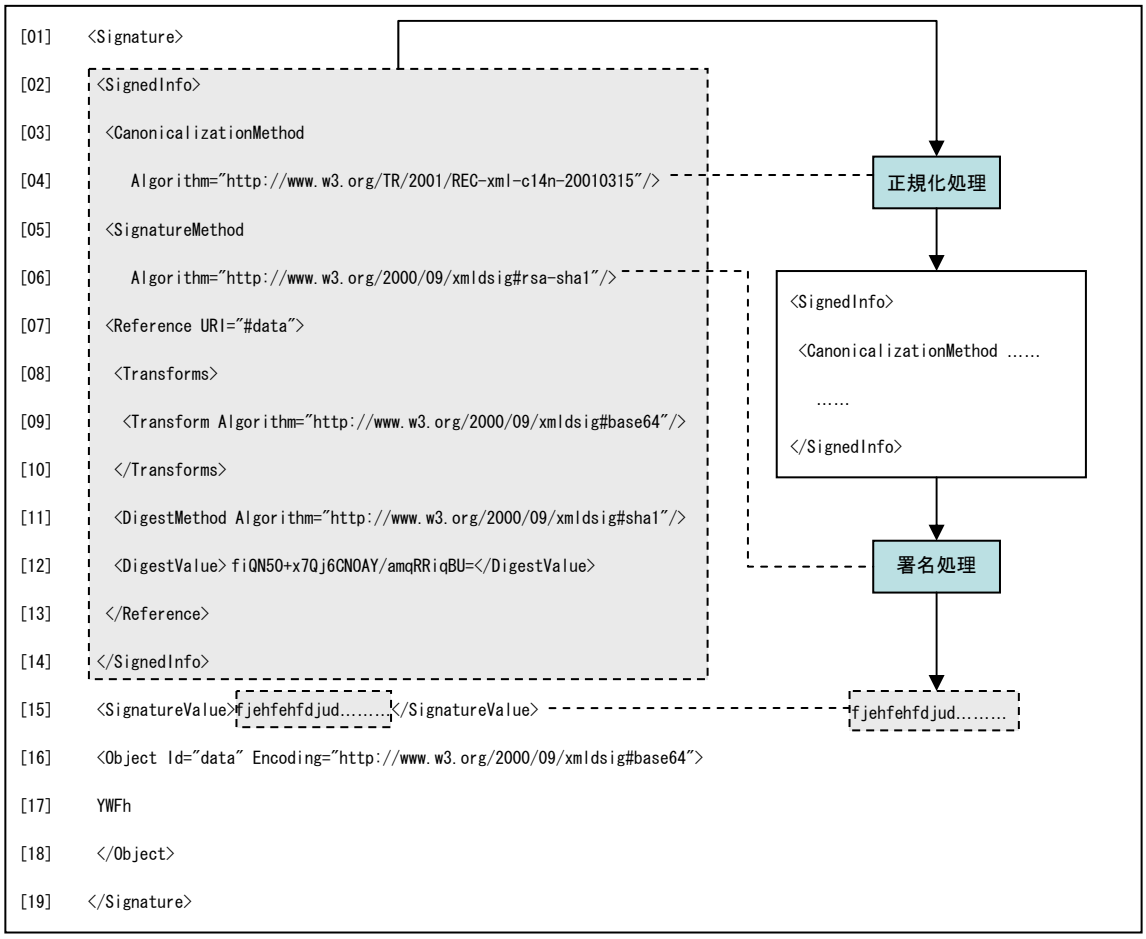


図 2.1.3.3: 署名値の計算の流れ

以下、これら XML 署名の各要素について説明を行う。なお、XML 署名の名前空間 URI は　http://www.w3.org/2000/09/xmldsig#　と定義されおり、名前空間接頭辞としてしばしば"ds"が使用される。以下、文中の名前空間接頭辞"ds"は特に断りの無い限り xmlns:ds="http://www.w3.org/2000/09/xmldsig#" が宣言されているものとして扱う。

2.1.4. Signature 要素

Signature 要素は XML 署名での最上位要素であり、XML 署名規格で規定される他の要素はすべて Signature 要素の子孫要素になる。Signature 要素は必須要素である SignedInfo 要素および SignatureValue 要素、任意要素である KeyInfo 要素および Object 要素で構成される。以下に Signature 要素の XML Schema 定義を示す（リスト 2.1.4.1）。

リスト 2.1.4.1: Signature 要素の XMLSchema 定義

```
<element name="Signature" type="ds:SignatureType"/>
<complexType name="SignatureType">
  <sequence>
    <element ref="ds:SignedInfo"/>
    <element ref="ds:SignatureValue"/>
    <element ref="ds:KeyInfo" minOccurs="0"/>
    <element ref="ds:Object" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="Id" type="ID" use="optional"/>
</complexType>
```

Note:XML 署名規格では Signature 要素の Id 属性は任意属性であるが、XAdES 規格において Signature 要素を参照する要件があるため、XAdES を生成する場合は Id 属性をあらかじめ作成または別途追加する必要がある。

2.1.5. SignedInfo 要素

SignedInfo 要素は署名値を計算する際に入力となる署名対象情報を格納する要素である。SignedInfo 要素は CanonicalizationMethod 要素、SignatureMethod 要素および 1 つまたは複数の Reference 要素から構成される。

以下に SignedInfo 要素の XML Schema 定義を示す（リスト 2.1.5.1）。

リスト 2.1.5.1: SignedInfo 要素の XMLSchema 定義

```
<element name="SignedInfo" type="ds:SignedInfoType"/>
<complexType name="SignedInfoType">
  <sequence>
    <element ref="ds:CanonicalizationMethod"/>
    <element ref="ds:SignatureMethod"/>
    <element ref="ds:Reference" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="Id" type="ID" use="optional"/>
</complexType>
```

2.1.5.1. CanonicalizationMethod 要素

XML は属性の並び方や空白文字、改行の扱い等文法上表現の自由度が高いという特徴があり、同じ意味を持つ XML 文書でも複数の異なる表現をとることができる。一方で、XML 署名で使用される署名アルゴリズムやダイジェストアルゴリズムは bit 単位で厳密に計算される。このため、同じ意味でありながら異なる表現を持つ 2 つの XML 文書に対しそのままアルゴリズムを適用すると、2 つの XML 文書が全く異なる文書として扱われてしまい、本来 XML が持つ表現の自由度の高さといった特徴を十分に活かせなくなる。この問題を解決するため、XML 署名では同じ意味をもつ XML 文書が bit 単位で同じ表現の

文書になるような変換処理を行う。この処理を正規化(canonicalization)と呼ぶ。

CanonicalizationMethod 要素は署名値の計算時に SignedInfo 要素に適用される正規化アルゴリズムを指定するための必須要素である。

正規化アルゴリズムは、必須属性である Algorithm 属性にて URI で指定する。

以下に CanonicalizationMethod 要素の XML Schema 定義を示す（リスト 2.1.5.1.1)。

リスト 2.1.5.1.1: CanonicalizationMethod 要素の XMLSchema 定義

```
<element
     name="CanonicalizationMethod" type="ds:CanonicalizationMethodType"/>
<complexType name="CanonicalizationMethodType" mixed="true">
  <sequence>
    <any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
    <!-- (0,unbounded) elements from (1,1) namespace -->
  </sequence>
  <attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>
```

主な正規化アルゴリズムおよびその URI を以下に示す（表 2.1.5.1.1)。

表 2.1.5.1.1: 主な正規化アルゴリズムに関する URI

正規化アルゴリズム	URI
Canonical XML	http://www.w3.org/TR/2001/REC-xml-c14n-20010315
Canonical XML with Comments	http://www.w3.org/TR/2001/REC-xml-c14n-20010315#WithComments
Exclusive Canonical XML	http://www.w3.org/2001/10/xml-exc-c14n#
Exclusive Canonical XML with Comments	http://www.w3.org/2001/10/xml-exc-c14n#WithComments

XML 署名規格においては Canonical XML が必須の正規化アルゴリズムとして指定されている。以下にその概要および実際の処理例を示す（リスト 2.1.5.1.2、図 2.1.5.1.1)。なお、正規化処理では octet stream または node-set が入力され、octet stream が出力される。(octet stream および node-set の概要は図 2.1.5.1.2 参照。)

リスト 2.1.5.1.2: Canonical XML の処理概要

・UTF-8 以外でエンコードされている場合は UTF-8 に変換される。
・改行は入力や構文解析前に LF(#xA)に正規化(normalize)される。
・属性値は妥当性検証プロセッサが行うように正規化(normalize)される。例えば属性値中の改行やタブなどのホワイトスペースは空白文字(#x20)に変換され CDATA 以外の型の場合は連続する空白文字は１つの空白文字に変換される。
・文字参照および実体参照は特殊文字を除き文字に置換される。(例:2→2)
・CDATA セクションは文字コンテンツに置換される。
・XML 宣言と DTD は削除される。
・空要素は開始タグと終了タグの組に変換される。例えば <a/>は<a>のように変換される。
・要素の外および開始タグと終了タグの中のホワイトスペースは正規化される。
・文字コンテンツ中のすべてのホワイトスペースは保持される。ただし改行の正規化での文字削除は除く。
・属性値の区切りは二重引用符を用いる。
・属性値と文字コンテンツの特殊文字は文字参照に置換される。(例: →
)
・各要素に余分な名前空間宣言があれば除去される。
・各要素でのデフォルト属性が追加される。
・各要素の名前空間宣言と属性は辞書的順番で並べ替えられる。
・コメントノードは削除される。

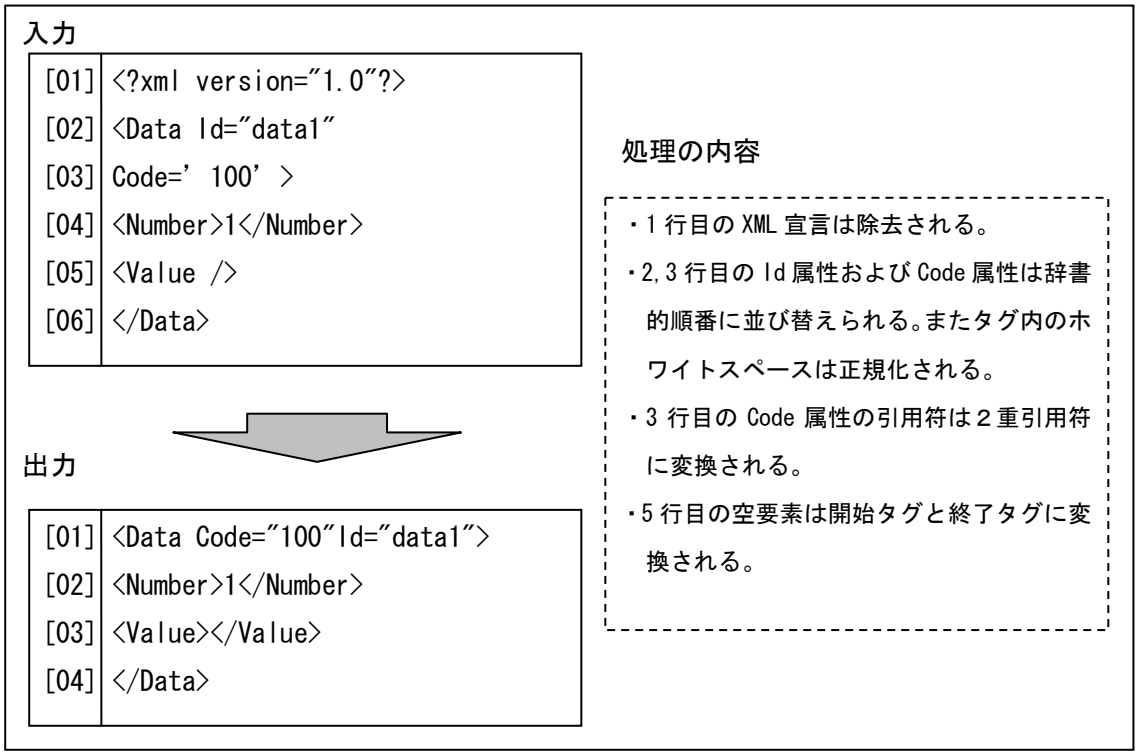


図 2.1.5.1.1: Canonical XML の処理例

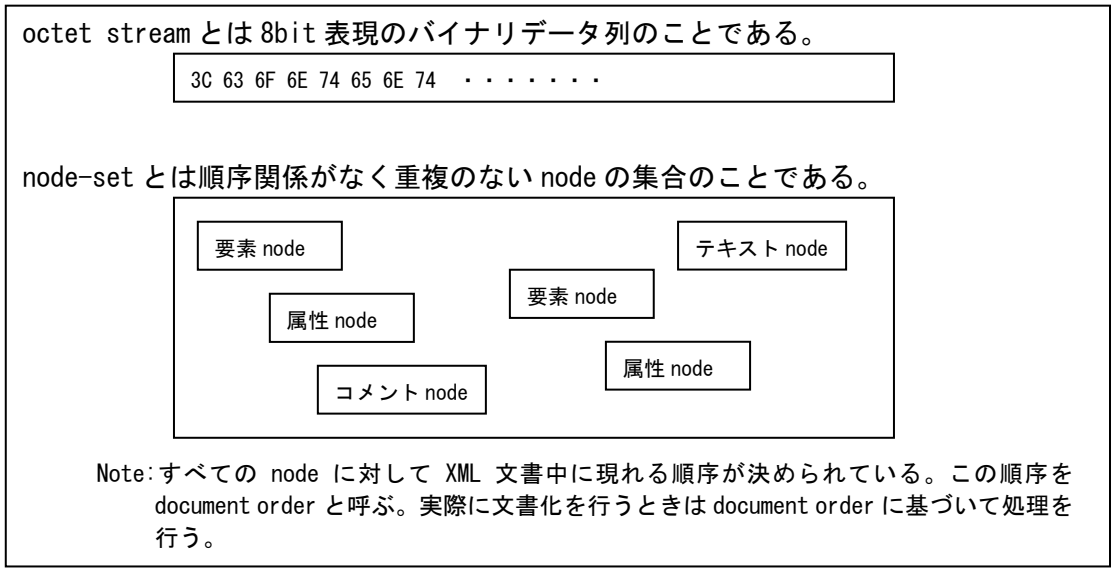


図 2.1.5.1.2: octet stream と node-set の概要

2.1.5.2. SignatureMethod 要素

SignatureMethod 要素は署名値を生成または検証するときに使用する署名アルゴリズムを指定する必須要素である。署名アルゴリズムは必須属性である Algorithm 属性にて URI で指定する。

以下に SignatureMethod 要素の XML Schema 定義を示す（リスト 2.1.5.2.1）。

リスト 2.1.5.2.1: SignatureMethod 要素の XMLSchema 定義

```
<element name="SignatureMethod" type="ds:SignatureMethodType"/>
<complexType name="SignatureMethodType" mixed="true">
  <sequence>
    <element name="HMACOutputLength"
             minOccurs="0" type="ds:HMACOutputLengthType"/>
    <any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
    <!-- (0,unbounded) elements from (1,1) external namespace -->
   </sequence>
 <attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>
```

主な署名アルゴリズムおよびその URI を以下に示す（表 2.1.5.2.1）。

表 2.1.5.2.1: 主な署名アルゴリズムに関する URI

署名アルゴリズム	URI
DSAwithSHA1(DSS)	http://www.w3.org/2000/09/xmldsig#dsa-sha1
RSAwithSHA1	http://www.w3.org/2000/09/xmldsig#rsa-sha1
RSAwithSHA256	http://www.w3.org/2001/04/xmldsig-more#rsa-sha256
RSAwithSHA384	http://www.w3.org/2001/04/xmldsig-more#rsa-sha384
RSAwithSHA512	http://www.w3.org/2001/04/xmldsig-more#rsa-sha512

2.1.5.3. Reference 要素

Reference 要素は署名対象となる要素またはデータの参照先、およびそのダイジェスト値を格納する要素である。任意要素である Transforms 要素、必須要素である DigestMethod 要素および DigestValue 要素にて構成される。URI 属性は署名対象となる要素またはデータを指定する任意属性である。Type 属性は署名対象データの種類についての情報が URI 形式で格納される任意属性である。なお Type 属性は補助的なものでありXML 署名規格では Type 属性値は検証には用いられない。

Note: XAdES 規格では特定の Type 属性が必須要件として規定されている。

以下に Reference 要素の XML Schema 定義を示す（リスト 2.1.5.3.1）。

リスト 2.1.5.3.1: Reference 要素の XMLSchema 定義

```
<element name="Reference" type="ds:ReferenceType"/>
<complexType name="ReferenceType">
  <sequence>
    <element ref="ds:Transforms" minOccurs="0"/>
    <element ref="ds:DigestMethod"/>
    <element ref="ds:DigestValue"/>
  </sequence>
  <attribute name="Id" type="ID" use="optional"/>
  <attribute name="URI" type="anyURI" use="optional"/>
  <attribute name="Type" type="anyURI" use="optional"/>
</complexType>
```

(1)参照処理モデル

Reference 要素の URI 属性での参照結果、後述する Transforms 要素による変換結果のデータ型は、octet stream または node-set になる。Transforms 要素での各変換処理は要求する入力形式が決まっている。実際の入力が異なっている場合には以下の処理を行う(リスト 2.1.5.3.2)。

リスト 2.1.5.3.2: 入力形式が異なる場合の一般的な処理の概要

- データが octet stream であるが次の変換が node-set の入力を要求している場合は、整形式の処理によって nose-set に構文解析することを試みなければならない。(MUST[32]4.3.3.2)
- データが node-set であり次の変換が octet stream の入力を要求している場合は Canonical XML を用いて octet stream に変換することを試みなければならない。(MUST[32]4.3.3.2)

URI 参照が同じ XML 文書中の要素への参照の場合、node-set を参照結果としなければならない(MUST[32]4.3.3.3)。参照結果はその参照要素および名前空間や属性を含むその要素のすべての子孫を加えたものとなる。URI 参照が例えば "#chapter1" のような barename 型 XPointer 表現で指定されている場合は、コメントノードを除いた node-set が参照結果となる。コメントノードを残したい場合は、'#xpointer(/)' や'#xpointer(id('ID'))' といった完全型の XPointer 表現で指定する必要がある。

URI 参照が Null URI(URI="")の場合は、その URI 属性が含まれる XML 文書のコメントノードを除いた全体の node-set を参照結果としなければならない(MUST[32]4.3.3.3)。外部リソースへの参照の場合、参照結果は octet stream にしなければならない(MUST[32]4.3.3.2)。また、外部の XML 文書への部分参照は URI 属性で指定するのではなく、後述する Transform 要素での XPath 変換を用いることが推奨されている

(RECOMMEND[32]4.3.3.2)。具体的な参照処理例を以下に示す(図 2.1.5.3.1、図 2.1.5.3.2、図 2.1.5.3.3、図 2.1.5.3.4)。

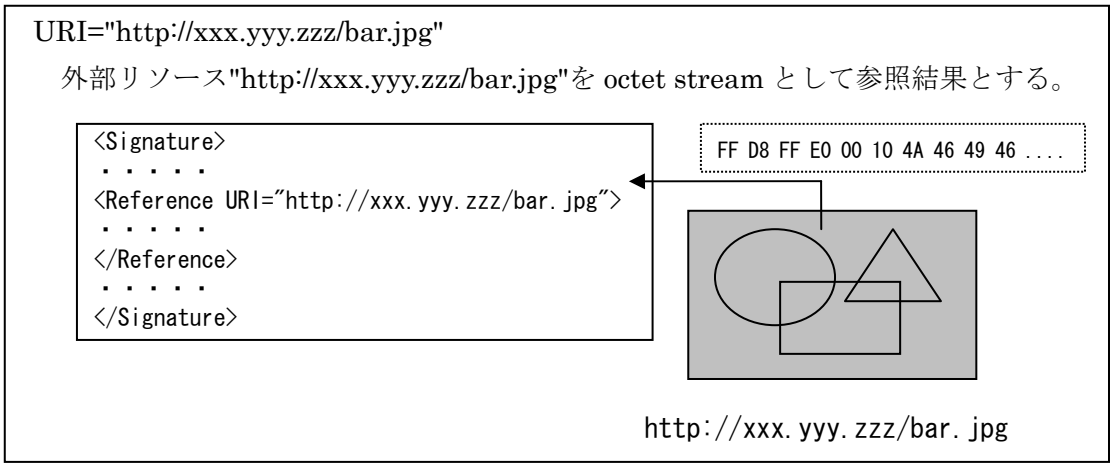


図 2.1.5.3.1: 参照処理例１（外部 JPEG ファイルへの参照）

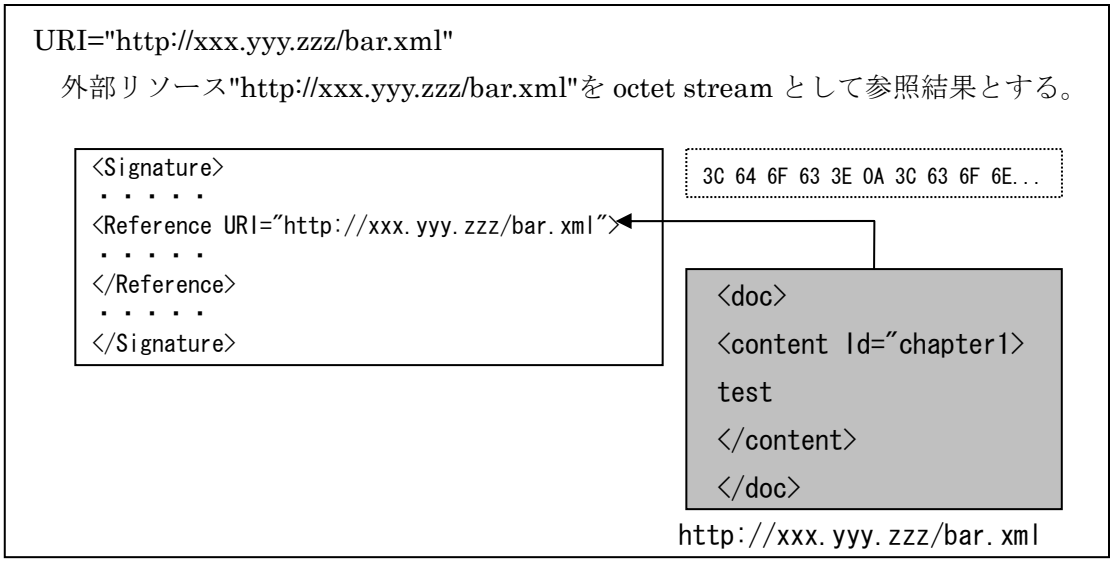


図 2.1.5.3.2: 参照処理例２（外部 XML ファイルへの参照）

URI="http://xxx.yyy.zzz/bar.xml#chapter1"

外部リソース"http://xxx.yyy.zzz/bar.xml"の chapter1 を Id 属性値としてもつ要素を octet stream として参照結果とする。ただし、互換性のため、XPath 変換を用いてこの要素を取得すべきである(SHOULD[32]4.3.3.2)。

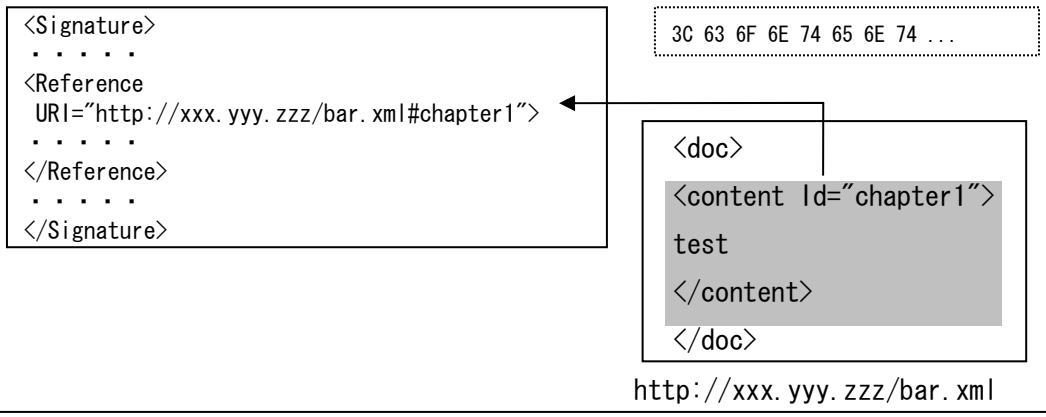


図 2.1.5.3.3: 参照処理例３（外部 XML ファイル内の要素への参照）

URI=""

署名が含まれる XML 文書のコメントノード以外の全体を node-set として参照結果とする。

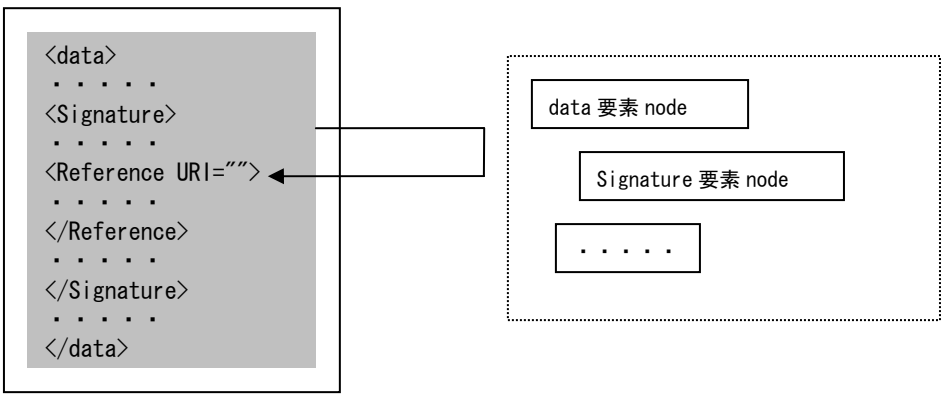


図 2.1.5.3.4: 参照処理例４（null URI の参照）

URI="#chapter1"

署名が含まれる XML 文書の chapter1 を Id 属性値としてもつ要素および名前空間や属性を含む（ただしコメントは含まない）その要素のすべての子孫を加えたものを node-set として参照結果とする。

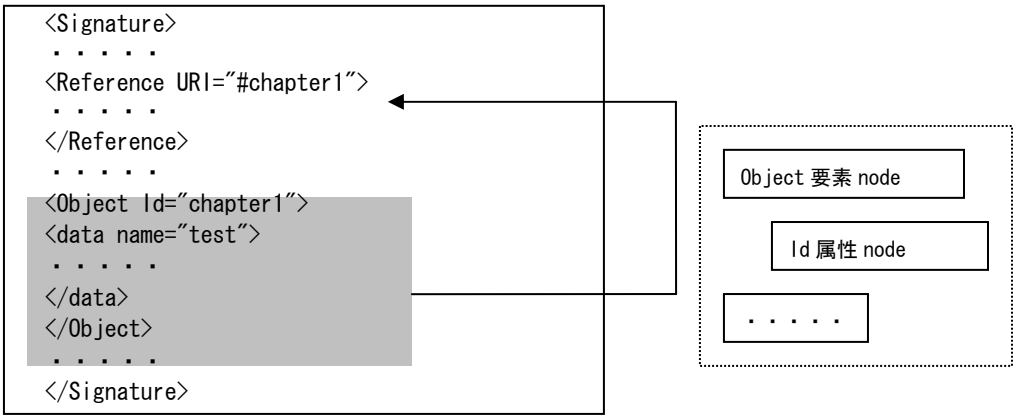


図 2.1.5.3.5: 参照処理例５（同じ XML 文書中の要素への参照）

(2)Transforms 要素

Transforms 要素は Reference 要素の URI 属性で参照した署名対象に対し、Base64 変換や正規化といった変換を指定する任意要素である。各変換方法は Transforms 要素の子要素である Transform 要素にて指定し、その変換アルゴリズムは Transform 要素の必須属性である Algorithm 属性にて URI で指定する。以下に Transforms 要素の XML Schema 定義を示す（リスト 2.1.5.3.3)。

リスト 2.1.5.3.3: Transforms 要素の XMLSchema 定義

```
<element name="Transforms" type="ds:TransformsType"/>
<complexType name="TransformsType">
  <sequence>
    <element ref="ds:Transform" maxOccurs="unbounded"/>
  </sequence>
</complexType>

<element name="Transform" type="ds:TransformType"/>
<complexType name="TransformType" mixed="true">
  <choice minOccurs="0" maxOccurs="unbounded">
    <any namespace="##other" processContents="lax"/>
    <!-- (1,1) elements from (0,unbounded) namespaces -->
    <element name="XPath" type="string"/>
  </choice>
  <attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>
```

Transform 要素はその並べられた順番の通りに変換処理が行わる。つまり n 番目の変換処理の入力は n-1 番目の変換処理の出力となる。また最初の変換処理の入力は Reference 要素の URI 属性での参照結果となり、最後の変換処理の出力は、後述するダイジェスト値の計算の入力となる。もし Transforms 要素が無い場合は、Reference 要素の URI 属性での参照結果がそのままダイジェスト値の計算の入力となる。Transform 要素で指定される主な変換アルゴリズムを以下に示す（表 2.1.5.3.1）。

表 2.1.5.3.1: 主な変換アルゴリズムに関する入出力および URI

変換アルゴリズム	URI
Base64	http://www.w3.org/2000/09/xmldsig#base64
XPath Filtering	http://www.w3.org/TR/1999/REC-xpath-19991116
Enveloped Signature Transform	http://www.w3.org/2000/09/xmldsig#enveloped-signature
XSLT Transform	http://www.w3.org/TR/1999/REC-xslt-19991116

※Transform 要素では上記の他に正規化変換も指定できる。正規化アルゴリズムおよびその URI については表 2.1.5.1.1 を参照。

以下それぞれの変換について概要を説明する。

・Base64 変換

この変換は Base64 デコードを行う変換処理である。octet stream が入力され、node-set が出力される。node-set が入力された場合は、以下の処理の出力を octet stream として入力とする（リスト 2.1.5.3.4)。

リスト 2.1.5.3.4: Base64 変換において node-set が入力された時の処理

1.XPath 変換 self::text() を適用する。 2.1 で出力された node-set の string-value を出力する

Base64 変換の具体例を以下に示す（図 2.1.5.3.6)。

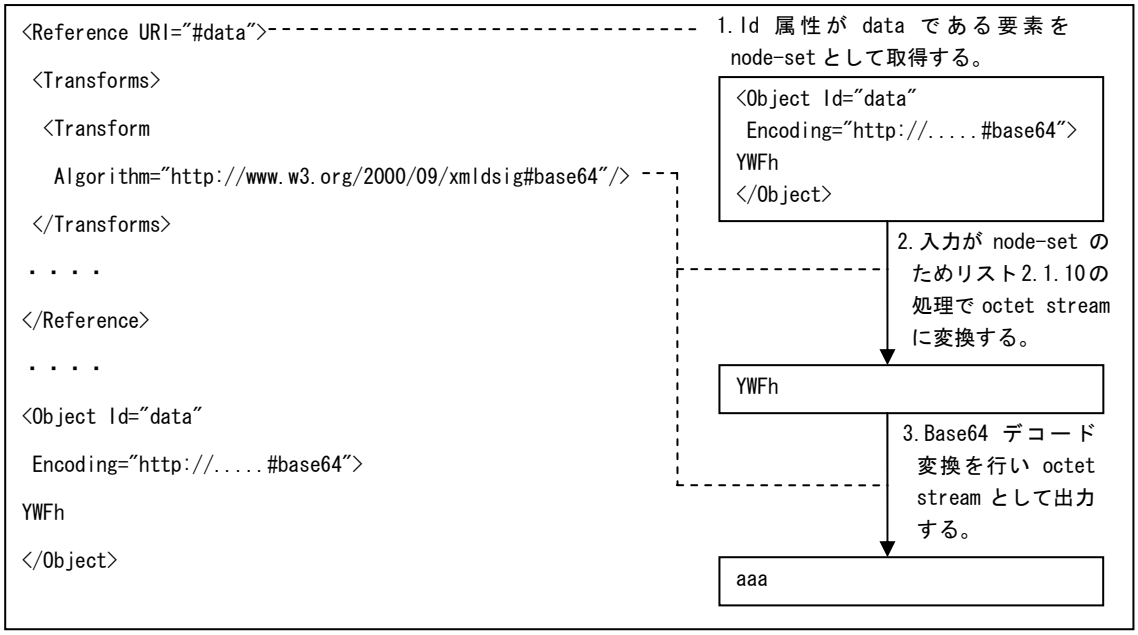


図 2.1.5.3.6: Base64 変換の例

・XPath filtering 変換

この変換は XPath 式に基づいた変換を行う。node-set が入力され node-set が出力される。octet stream が入力された場合は Canonical XML with Comment を使用するのに適した(コメントノードを除去していない)node-set に変換しなければならない（MUST[32]6.6.3)。Transform 要素の子要素である XPath 要素の内容として変換に用いる XPath 式が格納される。XPath 変換の具体例を以下に示す（図 2.1.5.3.7)。

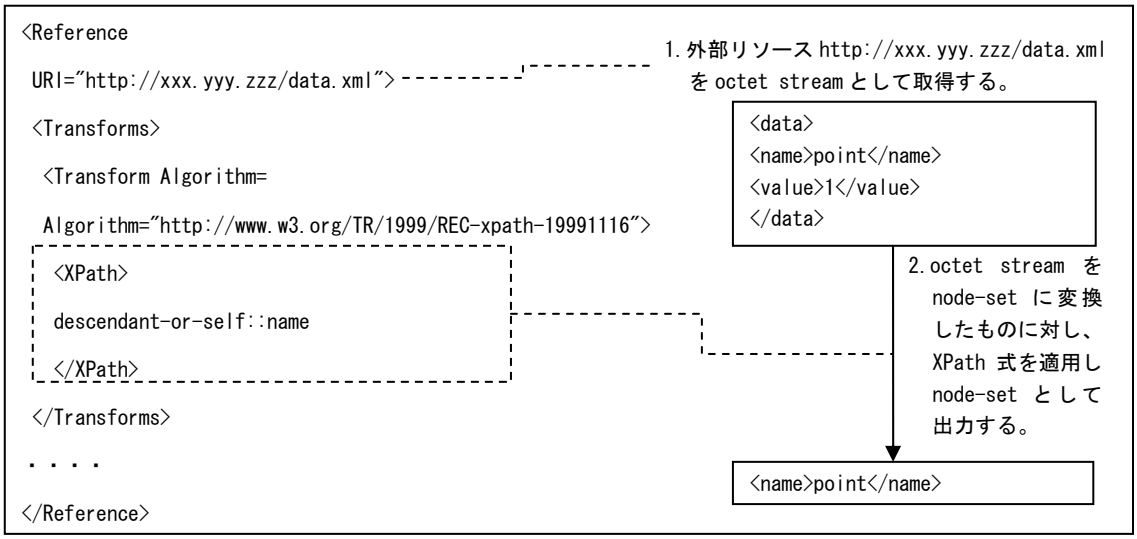


図 2.1.5.3.7: XPath 変換の例

・Enveloped Signature 変換

この変換は Enveloped Signature 変換自身が含まれる Signature 要素全体を取り除く変換である。node-set が入力され、node-set が出力される。

Enveloped Signature 変換の具体例を以下に示す（図 2.1.5.3.8)。

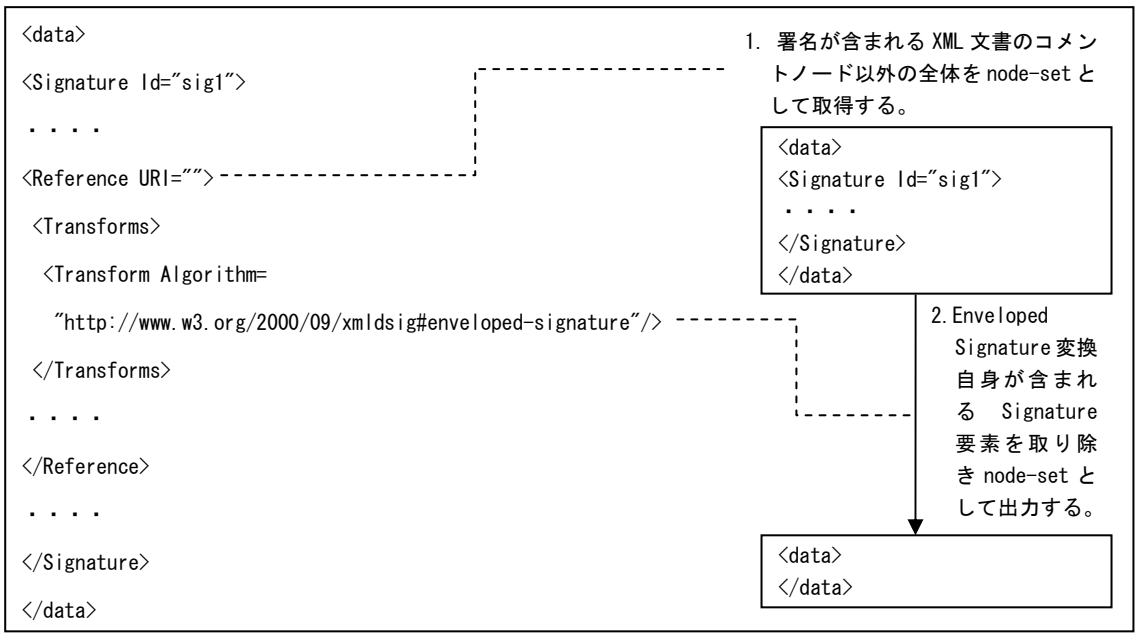


図 2.1.5.3.8: Enveloped Signature 変換の例

・XSLT 変換

この変換は XML スタイルシート変換を行う変換である。octet stream が入力され、octet stream が出力される。もし node-set が入力された場合はリスト 2.1.5.3.2 の処理により octet stream にしてから処理を行う。

適用される XML スタイルシート変換の内容は名前空間修飾されたスタイルシート変換の要素として示され、それは Transform 要素の唯一の子要素でなくてはならない(MUST[32]6.6.5)。XSLT 変換の具体例を以下に示す。

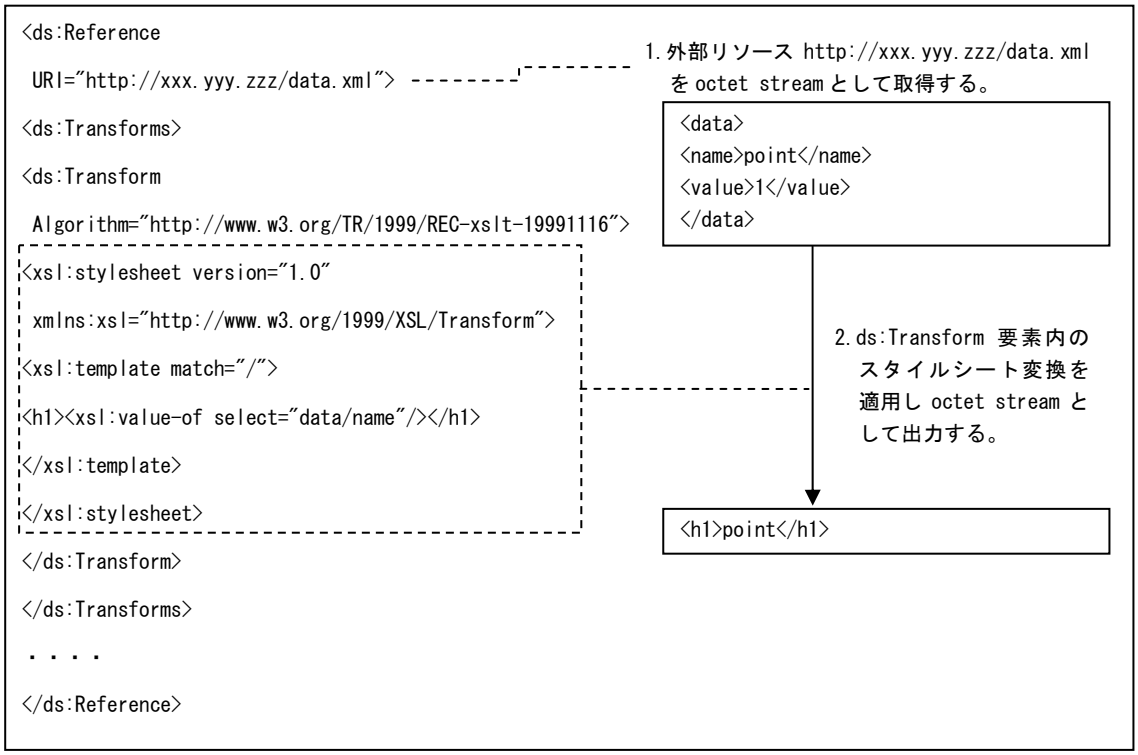


図 2.1.5.3.9: XSLT 変換の例

(3)DigestMethod 要素

DigestMethod 要素は Reference 要素の URI 属性による署名対象の参照結果、または Transforms 要素がある場合はその変換結果に適用するダイジェストアルゴリズムを指定する必須要素である。ダイジェストアルゴリズムは必須属性である Algorithm 属性にて URI で指定する。ダイジェスト値を計算する際の入力が node-set である場合は、リスト 2.1.5.3.2 で規定されているような octet stream への変換を行った結果が入力として使用され、入力が octet stream である場合はそのまま使用される。以下に DigestMehtod 要素の XML Schema 定義を示す（リスト 2.1.5.3.5)。

リスト 2.1.5.3.5: DigestMethod 要素の XMLSchema 定義

```
<element name="DigestMethod" type="ds:DigestMethodType"/>
<complexType name="DigestMethodType" mixed="true">
  <sequence>
    <any namespace="##other" processContents="lax"
         minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>
```

また、主なダイジェストアルゴリズムおよびその URI を以下に示す（表 2.1.5.3.2)。

表 2.1.5.3.2: 主なダイジェストアルゴリズムに関する URI

ダイジェスト アルゴリズム	URI
SHA-1	http://www.w3.org/2000/09/xmldsig#sha1
SHA-256	http://www.w3.org/2001/04/xmlenc#sha256
SHA-384	http://www.w3.org/2001/04/xmldsig-more#sha384
SHA-512	http://www.w3.org/2001/04/xmlenc#sha512

(4)DigestValue 要素

DigestValue 要素はダイジェスト値を格納するための要素である。この値は Base64 形式でエンコードされたものが使用される。

以下に DigestValue 要素の XML Schema 定義を示す。

リスト 2.1.5.3.6: DigestValue 要素の XMLSchema 定義

```
<element name="DigestValue" type="ds:DigestValueType"/>
<simpleType name="DigestValueType">
  <restriction base="base64Binary"/>
</simpleType>
```

2.1.6. SignatureValue 要素

SignatureValue 要素は署名値そのものを格納するための要素である。

実際には SignedInfo 要素に対し、CanonicalizationMethod で指定された正規化アルゴリズムを用いて正規化し、SignatureMethod 要素で指定された署名アルゴリズムを用いて算出された値を Base64 形式でエンコードしたものが格納される。

SignatureValue 要素の XML Schema 定義を以下に示す（リスト 2.1.6.1)。

リスト 2.1.6.1: SignatureValue 要素の XMLSchema 定義

```
<element name="SignatureValue" type="ds:SignatureValueType"/>
<complexType name="SignatureValueType">
  <simpleContent>
    <extension base="base64Binary">
      <attribute name="Id" type="ID" use="optional"/>
    </extension>
  </simpleContent>
</complexType>
```

2.1.7. KeyInfo 要素

KeyInfo 要素は署名検証に使用される鍵情報を格納する任意要素である。

XML 署名規格では鍵の信頼モデルについての制限はなく、X.509 公開鍵証明書(以下証明書とする)を利用する PKI 以外にも PGP(Pretty Good Privacy)や SPKI(Simple Public Key Infrastructure)などの鍵情報も格納できるように考慮されている。

以下に KeyInfo 要素の XML Schema 定義を示す（リスト 2.1.7.1)。

リスト 2.1.7.1: KeyInfo 要素の XMLSchema 定義

```
<element name="KeyInfo" type="ds:KeyInfoType"/>
<complexType name="KeyInfoType" mixed="true">
  <choice maxOccurs="unbounded">
    <element ref="ds:KeyName"/>
    <element ref="ds:KeyValue"/>
    <element ref="ds:RetrievalMethod"/>
    <element ref="ds:X509Data"/>
    <element ref="ds:PGPData"/>
    <element ref="ds:SPKIData"/>
    <element ref="ds:MgmtData"/>
    <any processContents="lax" namespace="##other"/>
    <!-- (1,1) elements from (0,unbounded) namespaces -->
  </choice>
  <attribute name="Id" type="ID" use="optional"/>
</complexType>
```

以下 KeyInfo 要素で使用される要素のうち、利用頻度が高くまた XAdES 規格中でも言及されている X509Data 要素について説明する。

2.1.7.1. X509Data 要素

X509Data 要素は 1 つまたは複数の証明書またはその識別子、および証明書失効リスト(以

下 CRL とする)に関する情報を格納するための任意要素である。

以下に X509Data 要素の XML Schema 定義を示す（リスト 2.1.7.1.1)。

リスト 2.1.7.1.1: X509Data 要素の XMLSchema 定義

```
<element name="X509Data" type="ds:X509DataType"/>
<complexType name="X509DataType">
  <sequence maxOccurs="unbounded">
    <choice>
      <element name="X509IssuerSerial" type="ds:X509IssuerSerialType"/>
      <element name="X509SKI" type="base64Binary"/>
      <element name="X509SubjectName" type="string"/>
      <element name="X509Certificate" type="base64Binary"/>
      <element name="X509CRL" type="base64Binary"/>
      <any namespace="##other" processContents="lax"/>
    </choice>
  </sequence>
</complexType>
<complexType name="X509IssuerSerialType">
  <sequence>
    <element name="X509IssuerName" type="string"/>
    <element name="X509SerialNumber" type="integer"/>
  </sequence>
</complexType>
```

X509Data 要素は以下の要素が少なくとも 1 つ含まれる。(同じ証明書に関する複数の情報がある場合と X509Data 要素中に複数の要素が含まれる場合とがある。)(リスト 2.1.7.1.2)

リスト 2.1.7.1.2: X509Data 要素の構成要素

・X509IssuerSerial 要素
　X509IssuerSerial 要素は、証明書の 発行者 DN とシリアル番号の組を RFC 2253[32]の表現に準拠した形で格納する要素である。発行者 DN は X509IssuerName 要素に格納され、シリアル番号は X509SerialNumber 要素に格納される。
・X509SubjectName 要素
　X509SubjectName 要素は、証明書の主体者 DN を RFC 2253 の表現に準拠した形で格納する要素である。
・X509SKI 要素
　X.509 V3 拡張の主体者鍵識別子の値を Base64 形式で格納する要素である。なお格納する値は DER エンコードした値ではなく、そのままの値を用いる
・X509Certificate 要素
　X509Certificate 要素には１つの証明書が Base64 形式でエンコードされた形で格納される。
・上記の要素に付随または補完する外部名前空間からの要素
・X509CRL 要素

また、これらの要素に対し以下の制約がある。

X509IssuerSerial 要素、X509SKI 要素、X509Subject 要素は検証のための鍵を含む証明書（すなわち署名者証明書）を参照するものでなくてはならない(MUST[32]4.4.4)。また、これらの要素は 1 つの証明書に対して 1 つの X509Data 要素中にまとめて存在しなければならず(MUST[32]4.4.4)、その証明書の実体が格納される場合にはこの同一の X509Data 要素中に存在しなければならない(MUST[32]4.4.4)。

同じ鍵であるが異なる証明書に関係する X509IssuerSerial 要素、X509SKI 要素、X509Subject 要素は、1 つの KeyInfo 要素にまとめられなければならない(MUST[32]4.4.4)。しかし、それらは複数の X509Data 要素に格納される場合もある(MAY[32]4.4.4)。

X509Data 要素に格納される証明書は、それ自身(署名者証明書)またはその検証鍵を含む証明書の証明書チェーンに関係するものでなくてはならない(MUST[32]4.4.4)。

複数の証明書および CRL を格納する場合は、1 つの X509Data 要素中に格納することもでき、また複数の X509Data 要素を用いることもできる。ただし 1 つの X509Data 要素中に複数の証明書および CRL を格納する場合、少なくとも 1 つは署名を検証するための鍵が含まれる証明書でなければならない(MUST[32]4.4.4)。以下に X509Data 要素の例を示す（リスト 2.1.7.1.3)。

リスト 2.1.7.1.3: X509Data 要素の例

```
[01] <KeyInfo>
[02]  <X509Data>
[03]   <X509IssuerSerial>
[04]     <X509IssuerName>
[05]      OU=SIGSUBCA, O=ECOM-TEST, C=JP
[06]     </X509IssuerName>
[07]     <X509SerialNumber>4949278979</X509SerialNumber>
[08]   </X509IssuerSerial>
[09]  </X509Data>
[10]  <X509Data>
[11]   <X509Certificate>MIICVWCCA...</X509Certificate>
[12]   <X509Certificate>MIICZVCCA...</X509Certificate>
[13]   <X509Certificate>MIICXTCCA...</X509Certificate>
[14]  </X509Data>
[15] </KeyInfo>
```

2～9 行目は X509IssuerSerial 要素で署名者証明書を示した例である。
10～14 行目は署名者証明書とその証明書チェーンを構成する証明書を格納した例である。

2.1.8. Object 要素

Object 要素は任意のデータを含むことができる任意要素である。XML 署名中に複数存在することもある。以下に Object 要素の XML Schema 定義を示す（リスト 2.1.8.1)。

リスト 2.1.8.1: Object 要素の XMLSchema 定義

```
<element name="Object" type="ds:ObjectType"/>
<complexType name="ObjectType" mixed="true">
  <sequence minOccurs="0" maxOccurs="unbounded">
    <any namespace="##any" processContents="lax"/>
  </sequence>
  <attribute name="Id" type="ID" use="optional"/>
  <attribute name="MimeType" type="string" use="optional"/>
  <attribute name="Encoding" type="anyURI" use="optional"/>
</complexType>
```

Encoding 属性は要素の内容のエンコード方法を特定するための任意属性である。主なエンコード方法およびその URI を以下に示す（表 2.1.8.1)。

表 2.1.8.1: 主なエンコード方法に関する URI

エンコード方法	URI
Base64	http://www.w3.org/2000/09/xmldsig#base64

MimeType 属性は要素の内容についての任意属性である。RFC 2045[32]にて定義される MIME type 文字列が属性値となる。なお、この属性は補助的なものであり、XML 署名規格においては検証には必要とされない。

Enveloping 署名形式では Object 要素が用いられ、その際に Reference 要素から参照されるために Id 属性が用いられる。ダイジェスト値を計算する時は、開始タグおよび終了タグを含む Object 要素全体が計算対象となる。

Object 要素の開始･終了タグをダイジェスト値の計算から除外したい場合は、Reference 要素にて要素の内容をデータそのものとして識別するか、または Object 要素のタグを削除するような変換を使用しなければならない(MUST[32]4.5)。

Note:以下のようなケースでは Object 要素のタグを除去することが望まれるかもしれない。

- 署名の有効性を維持した状態で、署名の内部にある署名対象データを署名の外部に移動させる場合、または署名の外部にある署名対象データを署名の内部に格納する場合。
- Object 要素の内容が元のバイナリ文書をエンコードしたものの場合に、元の bit 表現に対する署名となるようにしたい場合。

2.2. XAdES の基本フォーマットと基本構造

XAdES の基本となる署名フォーマットは、XML 署名の<ds:Object>要素内に必要な情報が追加された形をとる。また、長期署名の形式として以下のものが定義されている。

- XAdES-BES（Basic Electronic Signature）

最も基本的な署名の形式で、XML 署名と比較すると署名者証明書を署名対象として保護することを要件としている点が異なる。

- XAdES-EPES（Explicit Policy based Electronic Signature）

XAdES-BES の形式の要件に加えて署名ポリシを必須とする要件を追加したものである。署名ポリシとは、署名者と検証者がデジタル署名を有効とみなすための、署名の生成と検証に関する一連の規則を定めるものである。

- XAdES-T (Electronic signature with time)

XAdES-BES や XAdES-EPES に、デジタル署名の存在時刻を確定するために電子署名文書の署名値に対して TSA から取得したタイムスタンプトークンを追加したものである。

- XAdES-C（Electronic signature with complete validation data references）

XAdES-T に、署名者証明書の検証に使われる証明書チェーン上の全ての証明書（署名者証明書を除く）への参照を加えたものである。

・　XAdES-C（Electronic signature with complete validation data references）
XAdES-T に、署名者証明書の検証に使われる証明書チェーン上の全ての証明書（署名者証明書を除く）および失効情報への参照を加えたものである。

・　XAdES-X(type1, type2)(Extended signatures with time forms)
XAdES-C の形式の要件に加えて、検証情報や署名値、署名タイムスタンプに対するタイムスタンプを追加したものである。タイムスタンプの付与の仕方によって 2 種類の形式がある。

・　XAdES-X-L(type1, type2)(Extended long electronic signatures with time)
XAdES-C の形式の要件に加えて、証明書チェーン上のすべての証明書および失効情報を加えたものである。

・　XAdES-A(Archival electronic signatures
デジタル署名を長期保存するために、署名対象や署名値、証明書や失効情報、および既に付与されているタイムスタンプに対してさらにタイムスタンプを付与したものである。

図 2.2.1 に XAdES-BES の基本構造を示し、リスト 2.2.1 に XAdES-BES の基本フォーマットを示す。

XAdES-BES の基本構造

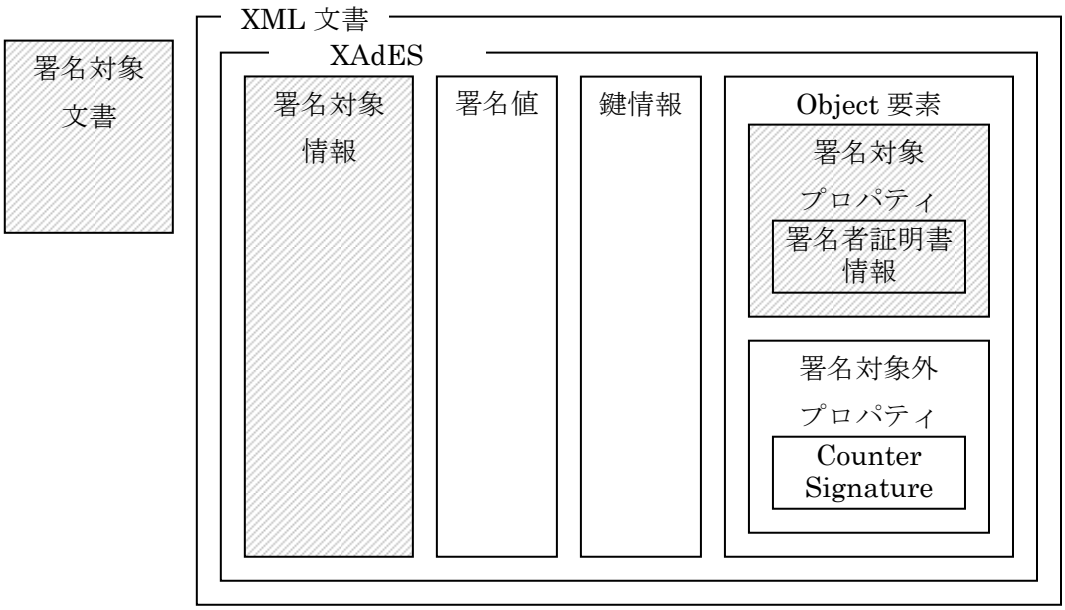
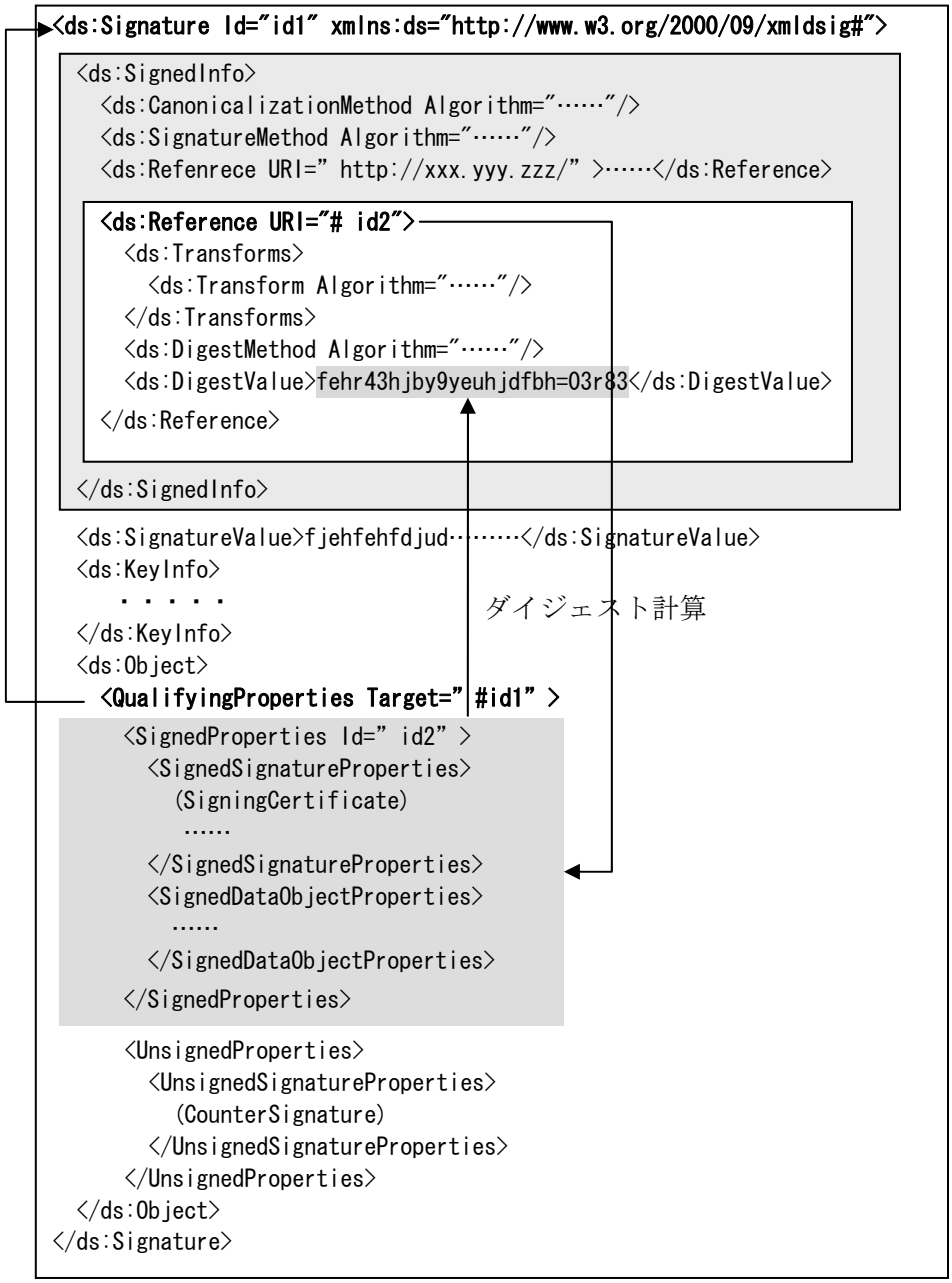


図 2.2.1　XAdES の基本構造の例：detached 形式の XAdES の基本構造の例を示す。

リスト 2.2.1：XAdES-BES の基本フォーマット



XAdES では、ds:Object 要素内の Qualifying Properties 要素に長期署名フォーマットに必要な、署名対象プロパティ（SignedProperty 要素）や署名対象外プロパティ（UnsignedProperty 要素）を格納する。2.2.1～2.2.7 に各要素の定義について説明する。

2.2.1. QualifyingProperties 要素

この要素は、<ds:Object>要素内に含められ長期保存に必要な要素を格納するコンテナの役割を果たす。リスト 2.2.1.1 に QualifyingProperties 要素の XMLSchema 定義を示す。

リスト 2.2.1.1：QualifyingProperties 要素の XMLSchema 定義

```
<xsd:element name="QualifyingProperties" type="QualifyingPropertiesType"/>
<xsd:complexType name="QualifyingPropertiesType">
  <xsd:sequence>
    <xsd:element name="SignedProperties"
                 type="SignedPropertiesType"   minOccurs="0"/>
    <xsd:element name="UnsignedProperties"
                 type="UnsignedPropertiesType" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="Target" type="xsd:anyURI" use="required"/>
  <xsd:attribute name="Id"     type="xsd:ID"     use="optional"/>
</xsd:complexType>
```

QualifyingProperties 要素には、署名値の計算対象となる SignedProperties 要素と、署名値の計算対象にはならない UnsignedProperties 要素の二つの要素が含まれる。また、SignedProperties 要素は必ず一つ存在しなくてはならない(MUST[22]6.2)。Target 属性は必須属性であり、ds:Signature 要素の Id 属性を参照する必要がある(MUST[22]6.2)。Id 属性は、オプションである。リスト 2.2.1.2 に QualifyingProperties 要素の例を示す。

リスト 2.2.1.2：QualifyingProperties 要素の例

```
<xa:QualifyingProperties Target="#Signature-ID1">
  <xa:SignedProperties Id="SignedProperties-ID1">
    <xa:SignedSignatureProperties>
        .........
    </xa:SignedSignatureProperties>
  </xa:SignedProperties>
  <xa:UnsignedProperties>
    <xa:UnsignedSignatureProperties>
        .........
    </xa:UnsignedSignatureProperties>
  </xa:UnsignedProperties>
</xa:QualifyingProperties>
```

2.2.2. SignedProperties 要素

この要素は、XML 署名の署名値の計算対象となるよう、XML 署名の中から ds:Reference 要素で参照される。この要素は一つだけ SignedSignatureProperties 要素を含まなくてはならず(MUST[22]6.2.1)、SignedDataObjectProperties 要素を含む場合もある。リスト 2.2.2.1 に SignedProperties 要素の XMLSchema 定義を示す。

リスト 2.2.2.1：SignedProperties 要素の XMLSchema 定義

```
<xsd:element name="SignedProperties" type="SignedPropertiesType" />
<xsd:complexType name="SignedPropertiesType">
  <xsd:sequence>
    <xsd:element name="SignedSignatureProperties"
                 type="SignedSignaturePropertiesType"/>
    <xsd:element name="SignedDataObjectProperties"
                 type="SignedDataObjectPropertiesType" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>
```

また、SignedProperties要素を参照しているds:Reference要素のType属性に以下の値をセットしなければならない(MUST[22]6.3.1)。

http://uri.etsi.org/01903#SignedProperties

2.2.3. UnsignedProperties 要素

この要素は、署名対象としない要素を持つ。リスト 2.2.3.1 に UnsignedProperties 要素の XMLSchema 定義を示す。

リスト 2.2.3.1：UnsignedProperties 要素の XMLSchema 定義

```
<xsd:element name="UnsignedProperties" type="UnsignedPropertiesType" />
<xsd:complexType name="UnsignedPropertiesType">
  <xsd:sequence>
    <xsd:element name="UnsignedSignatureProperties"
                 type="UnsignedSignaturePropertiesType" minOccurs="0"/>
    <xsd:element name="UnsignedDataObjectProperties"
                 type="UnsignedDataObjectPropertiesType" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>
```

2.2.4. SignedSignatureProperties 要素

この要素は、QualifyingProperties 要素の Target 属性で指定された XML 署名に関する要素を子孫要素に持ち、XML 署名の署名対象として署名計算に含められる。リスト 2.2.4.1 に SignedSignatureProperties 要素の XMLSchema 定義を示す。

リスト 2.2.4.1：SignedSignatureProperties 要素の XMLSchema 定義

```
<xsd:element name="SignedSignatureProperties"
             type="SignedSignaturePropertiesType" />

<xsd:complexType name="SignedSignaturePropertiesType">
  <xsd:sequence>
    <xsd:element name="SigningTime"
                 type="xsd:dateTime" minOccurs="0"/>
    <xsd:element name="SigningCertificate"
                 type="CertIDListType" minOccurs="0"/>
    <xsd:element name="SignaturePolicyIdentifer"
                 type="SignaturePolicyIdentifierType" minOccurs="0"/>
    <xsd:element name="SignatureProductionPlace"
                 type="SignatureProductionPlaceType"  minOccurs="0"/>
    <xsd:element name="SignerRole"
                 type="SignerRoleType" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>
```

また、リスト 2.2.4.2 に SignedSignatureProperties 要素の例を示す。

リスト 2.2.4.2：SignedSignatureProperties 要素の例

```
<xa:SignedSignatureProperties>
  <xa:SigningTime>2006-11-28T15:30:38+09:00</xa:SigningTime>
  <xa:SigningCertificate>
    <xa:Cert>
      <xa:CertDigest>
        <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
        <ds:DigestValue>kOKd7owyE8ADy5h43OnBbZQ=</ds:DigestValue>
      </xa:CertDigest>
      <xa:IssuerSerial>
        <ds:X509IssuerName>OU=ESA1, O=ECOM-TEST-ROOTCA, C=JP</ds:X509IssuerName>
        <ds:X509SerialNumber>3197562185174366618O</ds:X509SerialNumber>
      </xa:IssuerSerial>
    </xa:Cert>
  </xa:SigningCertificate>
</xa:SignedSignatureProperties>
```

2.2.5. UnsignedSignatureProperties 要素

この要素は、QualifyingProperties 要素の Target 属性で指定された XML 署名の持つ属性情報を子要素に持つが、XML 署名の署名対象として署名計算に含めない。リスト 2.2.5.1 に UnsignedSignatureProperties 要素の XMLSchema を示す。

リスト 2.2.5.1：UnsignedSignatureProperties 要素の XMLSchema 定義

```
<xsd:element name="UnsignedSignatureProperties"
             type="UnsignedSignaturePropertiesType"/>

<xsd:complexType name="UnsignedSignaturePropertiesType">
  <xsd:choice maxOccurs="unbounded">
    <xsd:element name="CounterSignature"
                 type="CounterSignatureType" />
    <xsd:element name="SignatureTimeStamp"
                 type="XAdESTimeStampType/>
    <xsd:element name="CompleteCertificateRefs"
                 type="CompleteCertificateRefsType"/>
    <xsd:element name="CompleteRevocationRefs"
                 type="CompleteRevocationRefsType"/>
    <xsd:element name="AttributeCertificateRefs"
                 type="CompleteCertificateRefsType"/>
    <xsd:element name="AttributeRevocationRefs"
                 type="CompleteRevocationRefsType"/>
    <xsd:element name="SigAndRefsTimeStamp"
                 type="XAdESTimeStampType"/>
    <xsd:element name="RefsOnlyTimeStamp"
                 type="XAdESTimeStampType"/>
    <xsd:element name="CertificateValues"
                 type="CertificateValuesType"/>
    <xsd:element name="RevocationValues"
                 type="RevocationValuesType"/>
    <xsd:element name="AttrAuthoritiesCertValues"
                  type="CertificateValuesType"/>
    <xsd:element name="AttributeRevocationValues"
                  Type="RevocationValuesType"/>
    <xsd:element name="ArchiveTimeStamp"
                 type="XAdESTimeStampType"/>
  </xsd:choice>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>
```

この要素には、以降で説明する様々な XAdES の形式で使われる長期保存に必要な要素が含められる。

2.2.6. SignedDataObjectProperties 要素

この要素は、署名対象文書(データ)となるデータオブジェクトの属性情報を表すような要素を子要素に持ち、署名値の計算で計算対象とされる要素である。リスト 2.2.6.1 に SignedDataObjectProperties 要素の XMLSchema 定義を示す。

リスト 2.2.6.1：SignedDataObjectProperties 要素の XMLSchema 定義

```
<xsd:element name="SignedDataObjectProperties"
             type="SignedDataObjectPropertiesType"/>

<xsd:complexType name="SignedDataObjectPropertiesType">
  <xsd:sequence>
    <xsd:element name="DataObjectFormat"
                 type="DataObjectFormatType"
                 minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="CommitmentTypeIndication"
                 type="CommitmentTypeIndicationType"
                 minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="AllDataObjectsTimeStamp"
                 type="XAdESTimeStampType"
                 minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="IndividualDataObjectsTimeStamp"
                 type="XAdESTimeStampType"
                 minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>
```

2.2.7. UnsignedDataObjectProperties 要素

この要素は、署名対象文書(データ)となるデータオブジェクトの属性情報を表すような要素を子要素に持ち、署名値の計算で計算対象とされない要素である。リスト 2.2.7.1 に UnsignedDataObjectProperties 要素の XMLSchema 定義を示す。

UnsignedDataObjectPropertiesについては、ETSI TS 101 703 CAdES V1.7.3[18]では、署名対象データの補足情報となるような非署名属性は定義されていないが、将来の拡張性と完全性の揺らぎを吸収するために定義している。

リスト 2.2.7.1：UnsignedDataObjectProperties 要素の XMLSchema 定義

```
<xsd:element name="UnsignedDataObjectProperties"
             type="UnsignedDataObjectPropertiesType" />

<xsd:complexType name="UnsignedDataObjectPropertiesType">
  <xsd:sequence>
    <xsd:element name="UnsignedDataObjectProperty"
                 type="AnyType" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>
```

2.2.8. XAdES-BES で利用するデータ型定義

XAdES では、いくつかのデータ型が定義されている。ここでは、XAdES-BES で利用するデータ型について説明する。

2.2.8.1. AnyType データ型

このデータ型は、要素の内容を特に規定したくない場合に用いる。このデータ型の要素の要素内容には、任意の要素やテキストなどを保持できる。また、任意の属性を制限なく追加することが出来る。リスト 2.2.8.1.1 に AnyType データ型の XMLShema 定義を示す。

リスト 2.2.8.1.1：AnyType データ型の XMLSchema 定義

```
<xsd:complexType name="AnyType" mixed="true">
  <xsd:sequence minOccurs="0" maxOccurs="unbounded">
    <xsd:any namespace="##any" processContents="lax"/>
  </xsd:sequence>
  <xsd:anyAttribute namespace="##any"/>
</xsd:complexType>
```

2.2.8.2. ObjectIdentifierType データ型

このデータ型は、オブジェクト識別子(OID)を格納するためのデータ型である。リスト 2.2.8.2.1 に ObjectIdentifierType データ型の XMLSchema 定義を示す。Identifier 要素では、ASN.1 におけるオブジェクトを識別する OID と XML のリソースを識別する URI の両方を指定することが出来る。

- XML リソースを指定する場合、Identifier 要素内に URI を記述する。Qualifier 属性は利用しない。
- ASN.1 で利用される OID でリソースを指定する場合は、URN の形式または URI としてエンコードした形で指定する。Qualifier 属性はどちらのエンコードが使われているかを示すために使われ、OIDAsURN、OIDAsURI のどちらかの値を取る。

Description 要素はオプションの要素で、オブジェクト識別子に関する説明文を格納する。また、DocumentationReferences 要素もオプションの要素で、オブジェクト識別子の追加説明への参照が任意の個数含まれる。

リスト 2.2.8.2.1：ObjectIdentifier データ型の XMLSchema 定義

```
<xsd:complexType name="ObjectIdentifierType">
  <xsd:sequence>
    <xsd:element name="Identifier" type="IdentifierType"/>
    <xsd:element name="Description" type="xsd:string" minOccurs="0"/>
    <xsd:element name="DocumentationReferences"
                 type="DocumentationReferencesType" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="IdentifierType">
  <xsd:simpleContent>
    <xsd:extension base="xsd:anyURI">
      <xsd:attribute name="Qualifier" type="QualifierType"  use="optional"/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
<xsd:simpleType name="QualifierType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="OIDAsURI"/>
    <xsd:enumeration value="OIDAsURN"/>
  </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="DocumentationReferencesType">
  <xsd:sequence maxOccurs="unbounded">
    <xsd:element name="DocumentationReference" type="xsd:anyURI"/>
  </xsd:sequence>
</xsd:complexType>
```

2.2.8.3. EncapsulatedPKIDataType データ型

このデータ型は、ASN.1 でエンコードされたデータを XML 文書に格納するために使われる。例えば、X.509 証明書や、失効リスト、属性証明書やタイムスタンプトークンのデータを格納するために使われる。格納時は、これらのデータを base64 でエンコードして格納する。Encoding 属性には、ASN.1 データのエンコード方法を URI で記述する。記述できる URI を表 2.2.8.3.1 に示す。リスト 2.2.8.3.1 に EncapsulatedPKIDataType データ型の XMLSchema 定義を示す。Encoding 属性が省略された場合は、DER エンコーディングでエンコードされたものとする。

表 2.2.8.3.1：ASN.1 データのエンコード方法に関する URI

エンコード方法	URI
DER	http://uri.etsi.org/01903/v1.2.2#DER
BER	http://uri.etsi.org/01903/v1.2.2#BER

CER	http://uri.etsi.org/01903/v1.2.2#CER
PER	http://uri.etsi.org/01903/v1.2.2#PER
XER	http://uri.etsi.org/01903/v1.2.2#XER

リスト 2.2.8.3.1：EncapsulatedPKIDataType データ型の XMLSchema 定義

```
<xsd:complexType name="EncapsulatedPKIDataType">
  <xsd:simpleContent>
    <xsd:extension base="xsd:base64Binary">
      <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
      <xsd:attribute name="Encoding" type="xsd:anyURI"
        use="optional"/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
```

2.3. XAdES の各形式の詳細

2.3.1. XAdES-BES (Basic electronic signature)

XAdES-BES は、ここまで説明してきた基本構造に基づき構成される。ここでは XAdES-BES に必要な要素について説明する。XAdES-BES では、署名者証明書が署名で保護されることが必須要件となっており、以下のどちらか一方が必須となる。

- <SignedSignatureProperties>要素の中に<SigningCertificate>要素を含み、それを署名値計算の対象として含める。
- <ds:Signature>要素に<ds:KeyInfo>が存在し、署名値計算の対象として含める。

以降、それぞれについて説明する。

2.3.1.1. SigningCertificate 要素

この要素には、署名者証明書のダイジェスト値と証明書への参照を含めなければならない。また、信頼点までの証明書チェーンを構成する証明書のダイジェスト値とシリアル番号を含むことも出来る。ただし、署名ポリシで規定されている場合には、信頼点までの証明書チェーンを構成する証明書のダイジェスト値と発行者のシリアル番号を含まなければならない(MUST[22]7.2.2)。リスト 2.3.1.1.1 に SigningCertificate 要素の XMLSchema 定義を示す。なお、ここで言う証明書の参照とは、証明書を発行した認証局の DN と証明書のシリアル番号を含む ds:X509IssuerSerialType 型の要素<IssuerSerial>で表現される。

この要素は、SigndProperties の含まれる他の要素とともに署名値の計算対象として署名計算に含められる。これらの要素は、Simple substitution 攻撃を防ぐために用いる。

リスト 2.3.1.1.1：SigningCertificate 要素の XMLSchema 定義

```
<xsd:element name="SigningCertificate" type="CertIDListType"/>

<xsd:complexType name="CertIDListType">
  <xsd:sequence>
    <xsd:element name="Cert" type="CertIDType" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CertIDType">
  <xsd:sequence>
    <xsd:element name="CertDigest" type="DigestAlgAndValueType"/>
    <xsd:element name="IssuerSerial" type="ds:X509IssuerSerialType"/>
  </xsd:sequence>
  <xsd:attribute name="URI" type="xsd:anyURI" use="optional"/>
</xsd:complexType>

<xsd:complexType name="DigestAlgAndValueType">
  <xsd:sequence>
    <xsd:element ref="ds:DigestMethod"/>
    <xsd:element ref="ds:DigestValue"/>
  </xsd:sequence>
</xsd:complexType>
```

リスト 2.3.1.1.2 に SigningCertificate 要素の例を示す。

リスト 2.3.1.1.2：SigningCertificate 要素の例

```
<xa:SigningCertificate>
  <xa:Cert>
    <xa:CertDigest>
      <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
      <ds:DigestValue>kOKd7owyE8ADy5h43OnBbZQ=</ds:DigestValue>
    </xa:CertDigest>
    <xa:IssuerSerial>
      <ds:X509IssuerName>OU=ESA1, O=ECOM-TEST-ROOTCA, C=JP</ds:X509IssuerName>
      <ds:X509SerialNumber>3197562185174366618O</ds:X509SerialNumber>
    </xa:IssuerSerial>
  </xa:Cert>
</xa:SigningCertificate>
```

2.3.1.2. ds:KeyInfo 要素

SigningCertificage 要素が存在しない、もしくは署名計算の対象となっていない場合は、ds:KeyInfo 要素が必須であり、以下の条件を満たさなければならない。

- ds:KeyInfo は、署名者証明書を含む ds:X509Data 要素を含まなければならない
- ds:KeyInfo は、信頼点までの証明書チェーンを構成する証明書も含む場合がある。
- ds:SignedInfo 要素の ds:Reference 要素で ds:KeyInfo を参照することにより、署名の計算対象として署名値の計算に含まれなければならない。

2.3.2. Explicit policy electronic signatures (XAdES-EPES)

XAdES-EPES は、XAdES で基本となる形式の 1 つで、XAdES-BES に署名ポリシに関する要素である SignaturePolicyIdentifier 要素を追加したものである。SignaturePolicyIdentifier 要素は、SignedSignatureProperties 要素に追加され XAdES-EPES では必須要素である。この属性は、署名対象要素であり署名で保護される。SignaturePolicyIdentifier 要素について以下に説明する。

2.3.2.1. SignaturePolicyIdentifier 要素

署名ポリシは、署名者と署名の検証者が守るべき一連の規則である。署名者は自分の意図と合致する署名ポリシを取得して署名に含め、検証者は検証処理の一環として署名ポリシを参照することにより、署名の意味や有効性を確認する。署名ポリシの記述形式には、人間が読んで理解できるフリーテキストによる形式と、ASN.1 や XML によるコンピュータによる処理が可能な形式とがある。署名ポリシには、次のような効果が期待される([43])。

(1)署名の意図や意味を明示し、電子署名の安全性を強化する。
(2)署名処理系の汎用利用が可能となる。
(3)長期署名の証明力を向上する。

リスト 2.3.2.1.1 に SignaturePolicyIdentifier 要素の XMLSchema 定義を示す。

リスト 2.3.2.1.1：SignaturePolicyIdentifier 要素の XMLSchema 定義

```
<xsd:element name="SignaturePolicyIdentifier"
               type="SignaturePolicyIdentifierType"/>

<xsd:complexType name="SignaturePolicyIdentifierType">
  <xsd:choice>
    <xsd:element name="SignaturePolicyId"
                   type="SignaturePolicyIdType"/>
    <xsd:element name="SignaturePolicyImplied"/>
  </xsd:choice>
</xsd:complexType>

<xsd:complexType name="SignaturePolicyIdType">
  <xsd:sequence>
    <xsd:element name="SigPolicyId"
                   type="ObjectIdentifierType"/>
    <xsd:element ref="ds:Transforms" minOccurs="0"/>
    <xsd:element name="SigPolicyHash"
                   type="DigestAlgAndValueType"/>
    <xsd:element name="SigPolicyQualifiers"
                   type="SigPolicyQualifiersListType"
                   minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="SigPolicyQualifiersListType">
  <xsd:sequence>
    <xsd:element name="SigPolicyQualifier"
                   type="AnyType"
                   maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

2.3.2.2. XAdES-BES でオプショナルな要素

XAdES-BES や XAdES-EPES 形式では、SingedSiganturePropertiesProperties 要素に以下の要素を含めることが出来る。

- SigningTime
- SignatureProductionPlace
- SignerRole

SignedDataObjectProperties 要素に以下の要素を含めることが出来る。

- DataObjectFormat
- CommitmentTypeIndication
- AllDataObjectsTimeStamp
- IndividualDataObjectsTimeStamp

UnsignedSignatureProperties 要素に以下の要素を含めることができる。

- CounterSignature

以降、それぞれを簡単に説明する。

2.3.2.3. SigningTime 要素

この要素は、署名者が署名を実行したと言明する保証されない時刻を表す。この要素に保管される時刻のソースは実装者に依存するのでタイムスタンプトークンで保証された時刻を表すとは限らない。内部のデータ型は、W3CRecommendation “XML Schema Part2:Datatypes”で定義される xsd:dateTime 型となる。この要素は、ひとつの XAdES 署名文書の中に 1 つだけ含めることができる。リスト 2.3.2.3.1 に SigningTime の XMLSchema 定義を示す。

リスト 2.3.2.3.1：SigningTime 要素の XMLSchema 定義

```
<xsd:element name="SigningTime" type="xsd:dateTime"/>
```

リスト 2.3.2.3.2 に SigningTime 要素の例を示す。

リスト 2.3.2.3.2：SigningTime 要素の例

```
<xa:SigningTime>2006-11-28T15:30:38+09:00</xa:SigningTime>
```

2.3.2.4. SignatureProductionPlace 要素

この要素は、署名が生成された場所を示す。リスト 2.3.2.4.1 に SignatureProductionPlace 要素の XMLSchema 定義を示す。

リスト 2.3.2.4.1：SignatureProductionPlace 要素の XMLSchema 定義

```
<xsd:element name="SignatureProductionPlace"
             type="SignatureProductionPlaceType"/>

<xsd:complexType name="SignatureProductionPlaceType">
  <xsd:sequence>
    <xsd:element name="City" type="xsd:string" minOccurs="0"/>
    <xsd:element name="StateOrProvince" type="xsd:string" minOccurs="0"/>
    <xsd:element name="PostalCode" type="xsd:string" minOccurs="0"/>
    <xsd:element name="CountryName" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
```

リスト 2.3.2.4.2 に SignatureProductionPlace 要素の例を示す。

リスト 2.3.2.4.2：SignatureProductionPlace 要素の例

```
<xa:SignatureProductionPlace>
  <xa:City>Tokyo</xa:City>
  <xa:CountryName>Japan</xa:CountryName>
</xa:SignatureProductionPlace>
```

2.3.2.5. SignerRole 要素

契約によっては、ある特定のポジションの人によって署名されたかどうかのみが重要である場合がある。この要素は、そのような場合に対応するために署名者の役割を表す要素である。リスト 2.3.2.5.1 に SignerRole 要素の XMLSchema 定義を示す。また、署名者の役割を記述するために以下の２つの方法を定義する。

- 署名者自らが主張する役割名
  ClaimedRoles 要素に役割名を含める
- 認定された役割を含む属性証明書
  CertifiedRolses 要素に属性証明書を格納する

リスト 2.3.2.5.1：SignerRole 要素の XMLSchema 定義

```
<xsd:element name="SignerRole" type="SignerRoleType"/>
<xsd:complexType name="SignerRoleType">
  <xsd:sequence>
    <xsd:element name="ClaimedRoles"
                 type="ClaimedRolesListType" minOccurs="0"/>
    <xsd:element name="CertifiedRoles"
                 type="CertifiedRolesListType" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ClaimedRolesListType">
  <xsd:sequence>
    <xsd:element name="ClaimedRole"
                 type="AnyType" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CertifiedRolesListType">
  <xsd:sequence>
    <xsd:element name="CertifiedRole"
                 type="EncapsulatedPKIDataType" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

2.3.2.6. DataObjectFormat 要素

この要素は、署名対象となるデータオブジェクトのデータフォーマットを記述する要素である。この要素は、もし署名されたデータ内に暗黙のうちにデータフォーマットが含まれておらず、署名されたデータを検証するため、人間のユーザにデータを提供する場合は必須である(SHOULD[22]7.2.5)。この要素は、署名対象となるデータオブジェクト毎に追加することができる。リスト 2.3.2.6.1 に DataObjectFormat 要素の XMLShema 定義を示す。

リスト 2.3.2.6.1：DataObjectFormat 要素の XMLSchema 定義

```
<xsd:element name="DataObjectFormat" type="DataObjectFormatType"/>

<xsd:complexType name="DataObjectFormatType">
  <xsd:sequence>
    <xsd:element name="Description"
                 type="xsd:string" minOccurs="0"/>
    <xsd:element name="ObjectIdentifier"
                 type="ObjectIdentifierType" minOccurs="0"/>
    <xsd:element name="MimeType"
                 type="xsd:string" minOccurs="0"/>
    <xsd:element name="Encoding"
                 type="xsd:anyURI" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="ObjectReference" type="xsd:anyURI" use="required"/>
</xsd:complexType>
```

ObjectReference 属性は必須属性であり、特定したいデータオブジェクトの対応する ds:Signature 要素内の ds:Reference 要素を参照する。その他に以下のような情報を伝達することが出来る。

- Description 要素で、署名されたデータオブジェクトに関するテキスト情報を記述できる。
- ObjectIdentifier 要素で、署名されたデータオブジェクトのタイプを OID で指定できる。
- MimeType 要素で、署名されたデータオブジェクトの MIME type を指定できる。
- Encoding 要素で、署名されたデータオブジェクトの enoding フォーマットを指定できる。

なお、DataObjectFomat 要素が存在する場合、Description 要素、ObjectIdentifier 要素および MimeType 要素のうち少なくとも 1 つが DataObjectFomat 要素内に存在しなくてはいけない(MUST[22]7.2.5)。

2.3.2.7. CommitmentTypeIndication 要素

リスト 2.3.2.7.1 に CommitmentTypeIndication 要素の XMLSchema 定義を示す。表 2.3.2.7.1 にコミットメントとその内容の一覧を示す。

リスト 2.3.2.7.1：CommitmentTypeIndication 要素の XMLSchema 定義

```
<xsd:element name="CommitmentTypeIndication"
             type="CommitmentTypeIndicationType"/>

<xsd:complexType name="CommitmentTypeIndicationType">
  <xsd:sequence>
    <xsd:element name="CommitmentTypeId"
                 type="ObjectIdentifierType"/>
    <xsd:choice>
      <xsd:element name="ObjectReference"
                   type="xsd:anyURI" maxOccurs="unbounded"/>
      <xsd:element name="AllSignedDataObjects"/>
    </xsd:choice>
    <xsd:element name="CommitmentTypeQualifiers"
                 type="CommitmentTypeQualifiersListType" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CommitmentTypeQualifiersListType">
  <xsd:sequence>
    <xsd:element name="CommitmentTypeQualifier"
                 type="AnyType" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

表 2.3.2.7.1：コミットメントの種別とその内容の一覧

コミットメント	内容/要素の値(URI)
Proof of origin	署名者がその文書を生成したこと、承認したこと、送信したことを示す
	http://uri.etsi.org/01903/v1.2.2#ProofOfOrigin.
Proof of receipt	署名者がその文書を受け取ったことを示す
	http://uri.etsi.org/01903/v1.2.2#ProofOfReceipt.
Proof of delivery	TSP(信頼できるサービスプロバイダがその文書をアクセスできる状態でローカルなストアに置いたことを提示したことを示す
	http://uri.etsi.org/01903/v1.2.2#ProofOfDelivery.
Proof of sender	その提示をしたエンティティがその文書を送信したことを示す（生成したものでなくてもよい）
	http://uri.etsi.org/01903/v1.2.2#ProofOfSender.
Proof of approval	署名者がその文書を承認したことを示す
	http://uri.etsi.org/01903/v1.2.2#ProofOfApproval.

Proof of creation	署名者がその文書を生成したことを示す（承認したり送信したりする必要はない）
	http://uri.etsi.org/01903/v1.2.2#ProofOfCreation.

2.3.2.8. AllDataObjectsTimeStamp 要素

この要素は、署名計算の実行前に署名対象データに付与されたタイムスタンプを格納する。リスト 2.3.2.8.1 に AllDataObjectsTimeStamp 要素の XMLSchema 定義を示す。

リスト 2.3.2.8.1：AllDataObjectsTimeStamp 要素の XMLSchema 定義

```
<xsd:element name="AllDataObjectsTimeStamp" type="XAdESTimeStampType"/>
```

ds:SignedInfo に含まれていて SignedProperties 要素以外に署名者が署名したい全ての ds:Referenece を元にタイムスタンプが計算される。この要素を生成するアプリケーションは、SignedPropeties 要素で参照しているものを除く全ての ds:Refernece 要素について Implicit Mode でハッシュ値を生成しなければならない(MUST[22]7.2.9)。

複数の異なる TSA からタイムスタンプを取得する場合、TSA 毎にこの要素を生成する。

2.3.2.9. IndividualDataObjectsTimeStamp 要素

この要素は、署名計算の実行前に個別のデータオブジェクトに対して付与されたタイムスタンプを格納する。リスト 2.3.2.9.1 に IndividualDataObjectsTimeStamp 要素の XMLSchema 定義を示す。

リスト 2.3.2.9.1：IndividualDataObjectsTimeStamp 要素の XMLSchema 定義

```
<xsd:element name="IndividualDataObjectsTimeStamp" type="XAdESTimeStampType"/>
```

ds:SignedInfo に含まれている任意の ds:Reference 要素を元にタイムスタンプが計算される。ただし、SignedProperties を参照している ds:Reference 要素を対象として含むことは出来ない。アプリケーションは、SignedPropeties 要素で参照しているものを除く ds:Refernece 要素でタイムスタンプを付与したいと署名対象について Include 要素を生成しなければならない(MUST[22]7.2.10)。また、各 Include 要素の referencedData 属性は ture でなくてはならない。ひとつの XAdES の文書の中にこの要素を複数含めることが出来る。

2.3.2.10. CounterSignature 要素

ある署名者の署名に対し、別の署名者が署名を付与したい場合には CounterSignature

要素を用いる。

・http://uri.etsi.org/01903#CountersignedSignature

Type 属性が上記の値を持つ ds:Reference 要素を含む XAdES 署名は、この要素で参照される署名の ConuterSignatue であることを示す。ConuterSignatue が実際にカウンタ署名された ds:SignatureValue 要素に署名するように、ds:Reference 要素は構築されなければならない(MUST[22]7.2.4)。全ての XML 署名規則は、前記の ds:Reference 要素の処理に適用される。リスト 2.3.2.10.1 に CounterSignature 要素の XMLSchema 定義を示す。

リスト 2.3.2.10.1：CounterSignature 要素の XMLSchema 定義

```
<xsd:element name="CounterSignature" type="CounterSignatureType" />

<xsd:complexType name="CounterSignatureType">
  <xsd:sequence>
    <xsd:element ref="ds:Signature"/>
  </xsd:sequence>
</xsd:complexType>
```

2.3.3. XAdES-T (Electronic signature with time)

XAdES-T は、デジタル署名の存在時刻を確定するために電子署名文書の署名値（ds:Signature 要素内の ds:SignedInfo 要素内の SignatureValue 要素）に対して TSA から取得したタイムスタンプトークンを追加したものである。署名値から生成したタイムスタンプトークンは、署名の存在時刻とともに、間接的に署名対象のデータオブジェクトの存在時刻も証明する。XAdES-T の形式は、XAdES-BES または XAdES-EPES の UnSignedSignatureProperties 要素に SignatureTimeStamp 要素を追加した形式である（UnsignedSignatureProperties 要素の XMLShema 定義はリスト 2.2.5.1 を参照のこと）。

図 2.3.3.1 に XAdES-T の基本構造を示す。

XAdES-T の基本構造

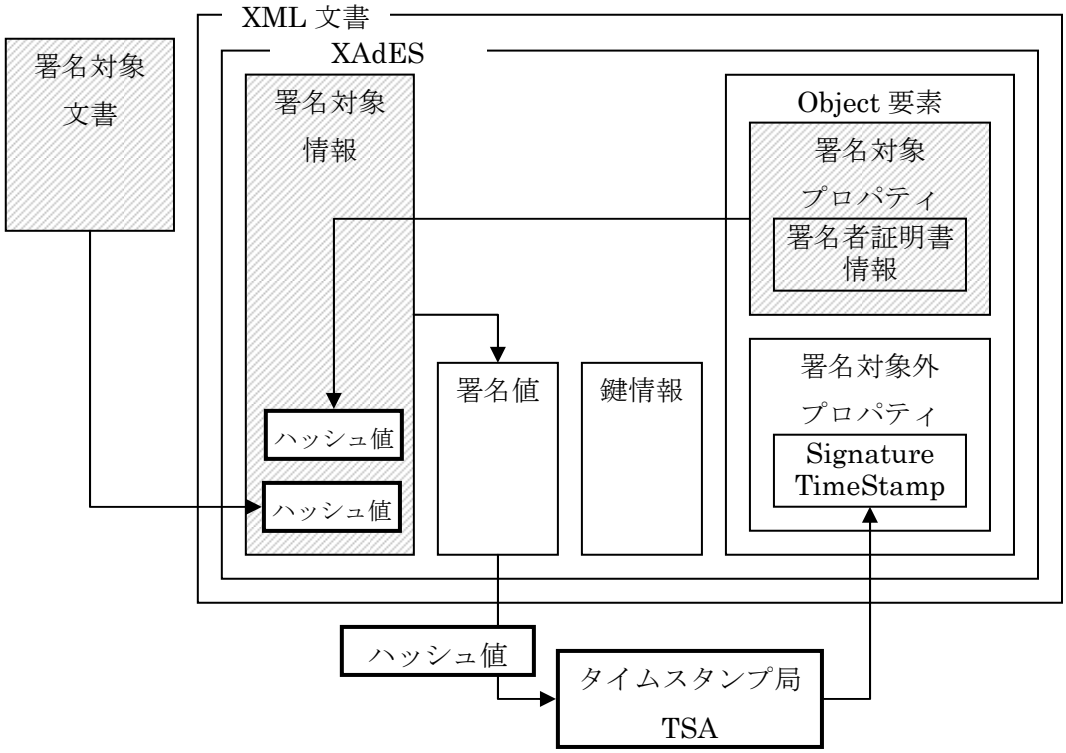


図 2.3.3.1：XAdES-T の基本構造

2.3.3.1. SignatureTimeStamp 要素

この要素は、ds:SignatureValue 要素に対して付与したタイムスタンプを格納する。1 つの署名に対して複数の異なった TSA から取得したタイムスタンプ格納するために、1 つの XAdES 文書に複数の SignatureTiemStamp 要素を格納することが出来る。リスト 2.3.3.1.1 に SignatureTimeStamp 要素の XMLSchema 定義を示す。

リスト 2.3.3.1.1：SignatureTimeStamp 要素の XMLSchema 定義

```
<xsd:element name="SignatureTimeStamp" type="XAdESTimeStampType"/>
```

リスト 2.3.3.1.2 に SignatureTimeStamp 要素の例を示す。

リスト 2.3.3.1.2：SignatureTimeStamp 要素の例

```
<xa:SignatureTimeStamp Id="STS-ID">
  <xa:EncapsulatedTimeStamp Encoding="http://uri.etsi.org/01903/v1.3.1#DER">+1hWZ
  cDX1yAAA=</xa:EncapsulatedTimeStamp>
</xa:SignatureTimeStamp>
```

この要素に含まれるタイムスタンプトークンの計算は、Implicit Mode（後述）で実行される。具体的には、ds:SignatureValue 要素が、TSA に送付されるダイジェスト値の計算の入力となる。

タイムスタンプトークン自体の検証情報（認証パス及び失効情報）は、次のいずれかに格納する。

1) タイムスタンプトークン内に含める
2) 署名者証明書の検証情報と同じ場所（CertificateValues, RevocationValues）

2.3.3.2. XAdESTimeStampType データ型

タイムスタンプを格納するデータ XAdESTimeStampType データ型は GenericTimeStampType データ型の派生として定義される。GenericTimeStampType データ型からは、XAdESTimeStampType データ型と OtherTimeStampType データ型が派生して定義されるが、OtherTimeStampType データ型は非推奨である。リスト 2.3.3.2.1 に GenericTimeStampType データ型の XMLSchema 定義を示す。また、それから派生される XAdESTimeStampType データ型の XMLSchema 定義をリスト 2.3.3.2.2 に示す。

リスト 2.3.3.2.1：GenericTimeStampType データ型の XMLSchema 定義

```
<xsd:element name=" Include"  type=" IncludeType" >
<xsd:complexType name="IncludeType">
  <xsd:attribute name="URI" type="xsd:anyURI" use="required"/>
  <xsd:attribute name="referencedData" type="xsd:boolean" use="optional"/>
</xsd:complexType>

<xsd:element name="ReferenceInfo" type="ReferenceInfoType"/>
<xsd:complexType name="ReferenceInfoType">
  <xsd:sequence>
    <xsd:element ref="ds:DigestMethod"/>
    <xsd:element ref="ds:DigestValue"/>
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
  <xsd:attribute name="URI" type="xsd:anyURI" use="optional"/>
</xsd:complexType>

<xsd:complexType name="GenericTimeStampType" abstract="true">
  <xsd:sequence>
    <xsd:choice minOccurs="0">
      <xsd:element ref="Include" maxOccurs="unbounded"/>
      <xsd:element ref="ReferenceInfo" maxOccurs="unbounded"/>
    </xsd:choice>
    <xsd:element ref="ds:CanonicalizationMethod" minOccurs="0"/>
    <xsd:choice maxOccurs="unbounded">
      <xsd:element name="EncapsulatedTimeStamp"
                   type="EncapsulatedPKIDataType"/>
      <xsd:element name="XMLTimeStamp" type="AnyType"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>
```

リスト 2.3.3.2.2：XAdESTimeStampType データ型の XMLSchema 定義

```
<xsd:element name="XAdESTimeStamp" type="XAdESTimeStampType"/>

<xsd:complexType name="XAdESTimeStampType">
  <xsd:complexContent>
    <xsd:restriction base="GenericTimeStampType">
      <xsd:sequence>
        <xsd:element ref="Include"  minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="ds:CanonicalizationMethod" minOccurs="0"/>
        <xsd:choice maxOccurs="unbounded">
          <xsd:element name="EncapsulatedTimeStamp"
                       type="EncapsulatedPKIDataType"/>
          <xsd:element name="XMLTimeStamp" type="AnyType"/>
        </xsd:choice>
      </xsd:sequence>
      <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

XAdESTimeStampType データ型の要素にタイムスタンプトークンが含まれ､TSA に送るダイジェスト値の算出方法に関して 2 通りの方法を定義する

Explicit　このメカニズムは、Include 要素を使用し、TSA に送るダイジェスト値の計算対象となるデータオブジェクトを参照する。

Implicit　Include 要素が存在しない場合、TSA に送るダイジェスト値の計算対象となるデータオブジェクトは暗黙的に参照される｡以下の要素はこのメカニズムで計算される。

・AllDataObjectsTimeStamp 要素
・IndividualDataObjectsTimeStamp 要素
・SignatureTimeStamp 要素
・SigAndRefsTimeStamp 要素
・Ref s OnlyTimeStamp 要素
・ArchiveTimeStamp 要素

以下は Explicit メカニズムに適応される。

Include メカニズム

Include 要素はタイムスタンプの付与されたデータオブジェクトを特定する。これらの出現順はデータオブジェクトがどのようにダイジェスト計算への入力の生成において、

どのように与えられるかを示す。次のタイムスタンプトークンコンテナがこのメカニズムを利用する。

1) IndividualDataObjectsTimeStamp.この場合各々の Include 要素は XAdES 署名における ds:Reference 要素のうちの 1 つへの URI を含む。

2) SigAndRefsTimeStamp 、 RefsOnlyTimeStamp および ArchiveTimeStamp のみ、およびこれらのエレメントとそれらタイムスタンプ対象である非署名属性のいくつかに同じ親を持っていない場合。(それらのいくつかは異なる QualifyingProperties 要素の中にあり、XAdES 署名は QualifyingPropertiesReference 要素を含む)

ds:Reference 要素で保護するタイムスタンプにおいて、referenceData 属性は存在するかもしれない。その値が true ならば、XML 署名の処理モデルに従って、ds:Reference 要素の処理した結果を元にタイムスタンプを計算する。もし属性が存在しないか、値が false ならば、ds:Reference 要素自身を計算する。タイムスタンプコンテナプロパティが存在するとき、それぞれの Include 要素は、以下に詳述される手順で処理される。

1. URI 属性で参照されているデータオブジェクトを取り出す。
2. 取り出されたデータが ds:Reference 要素で、referenceData 属性が true ならば、XML 署名の処理モデルによって処理された ds:Reference 要素を処理し結果を得る。そうでなければ、ds:Reference 要素自体を取り出す。
3. 取り出した結果のデータが XML 形式ならば正規化する。ds:Canonicalization が存在するならば、その要素で指示されたアルゴリズムを利用する。存在しないならば、XML 署名によって指定される標準的な正規化手法が使われる。
4. .処理されたオクテットを連結する。

2.3.4. Electronic signature with complete validation data references (XAdES-C)

XAdES-C は、XAdES-T に CompleteCertificateRefs 要素と CompleteRevocationRefs 要素を加えたものとなる。CompleteCertificateRefs 要素は、署名者証明書の検証に使われる証明書チェーン上の全ての証明書（署名者証明書を除く）への参照をもつ。

CompleteRevocationRefs 要素は、署名者証明書および CA 証明書の検証に必要な全ての失効情報への参照が格納される。なお、後述する XAdES-A を構成するときは、CompleteCertificateRefs, CompleteRevocationRefs はオプション要素となる。

以降、XAdES-C で使用する要素を順次説明する。

2.3.4.1. CompleteCertificateRefs 要素

この要素は、XAdES 文書中にひとつだけ含めることが出来る。リスト 2.3.4.1.1 に CompleteCertificateRefs 要素の XMLSchema 定義を示す。

リスト 2.3.4.1.1：CompleteCertificateRefs 要素の XMLSchema 定義

```
<xsd:element name="CompleteCertificateRefs"
             type="CompleteCertificateRefsType"/>

<xsd:complexType name="CompleteCertificateRefsType">
  <xsd:sequence>
    <xsd:element name="CertRefs" type="CertIDListType" />
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>
```

2.3.4.2. CompleteRevocationRefs 要素

リスト 2.3.4.2.1 に CompleteRevocationRefs 要素の XMLSchema 定義を示す。

リスト 2.3.4.2.1：CompleteRevocationRefs 要素の XMLSchema 定義

```
<xsd:element name="CompleteRevocationRefs" type="CompleteRevocationRefsType"/>

<xsd:complexType name="CompleteRevocationRefsType">
  <xsd:sequence>
    <xsd:element name="CRLRefs" type="CRLRefsType" minOccurs="0"/>
    <xsd:element name="OCSPRefs" type="OCSPRefsType" minOccurs="0"/>
    <xsd:element name="OtherRefs" type="OtherCertStatusRefsType"
                 minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>

<xsd:complexType name="CRLRefsType">
  <xsd:sequence>
    <xsd:element name="CRLRef" type="CRLRefType" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CRLRefType">
  <xsd:sequence>
    <xsd:element name="DigestAlgAndValue" type="DigestAlgAndValueType"/>
    <xsd:element name="CRLIdentifier" type="CRLIdentifierType" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CRLIdentifierType">
  <xsd:sequence>
    <xsd:element name="Issuer" type="xsd:string"/>
    <xsd:element name="IssueTime" type="xsd:dateTime" />
    <xsd:element name="Number" type="xsd:integer"  minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="URI" type="xsd:anyURI" use="optional"/>
</xsd:complexType>

<xsd:complexType name="OCSPRefsType">
  <xsd:sequence>
    <xsd:element name="OCSPRef" type="OCSPRefType" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="OCSPRefType">
  <xsd:sequence>
    <xsd:element name="OCSPIdentifier" type="OCSPIdentifierType"/>
    <xsd:element name="DigestAlgAndValue"
      type="DigestAlgAndValueType"
      minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
```

```
<xsd:complexType name="OCSPIdentifierType">
  <xsd:sequence>
    <xsd:element name="ResponderID" type="xsd:string"/>
    <xsd:element name="ProducedAt" type="xsd:dateTime"/>
  </xsd:sequence>
  <xsd:attribute name="URI" type="xsd:anyURI" use="optional"/>
</xsd:complexType>

<xsd:complexType name="OtherCertStatusRefsType">
  <xsd:sequence>
    <xsd:element name="OtherRef" type="AnyType"
      maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

ETSI TS 101 903 V1.3.2(2006-03)では、署名の中に属性証明書が含まれる場合は、AttributeCertifcateRefs 要素と AttributeRevocationRefs 要素も加えることが出来るように定義されている。

2.3.5. Extended signatures with time forms (XAdES-X)

XAdES-X は、将来の CA の危殆化に備えたり、検証データの完全性を確保したり検証データが入手困難になることに備えるため、XAdES-C を拡張したものである。拡張の仕方により 2 種類プロファイルを定義する。

なお、後述する XAdES-A 形式を構成する場合、XAdES-X として追加される SigAndRefsTimeStamp 要素および RefsOnlyTimeStamp 要素はオプション要素である。

- XAdES-X type1
  XAdES-C 全体に対するタイムスタンプを取得して追加するもので、具体的には UnsingedSignatureProperties 要素に SigAndRefsTimeStamp 要素を追加したものである（UnsignedSignatureProperties 要素の XMLSchema 定義はリスト 2.2.5.1 を参照のこと）。複数の TSP からタイムスタンプを取得する考慮して、複数の SigAndRefsTimeStamp 要素を追加することができるよう定義する。タイムスタンプを取得するために TSP に送信するハッシュ値は、SignatureValue 要素、SignatureTimeStamp 要素、CompleteCertificateRefs 要素および CompleteRevocationRefs 要素を元に計算する。

- XAdES-X type2
  検証情報のリファレンスのみに対するタイムスタンプを取得して追加するもので、具体的には UnsignedSignatureProperties 要素に RefsOnlyTimeStamp 要素を追加した

ものである（UnsignedSignatureProperties 要素の XMLSchema 定義はリスト 2.2.5.1 を参照のこと）。複数の TSP からタイムスタンプを取得する考慮して、複数の RefsOnlyTimeStamp 要素を追加することができるよう定義する。タイムスタンプを取得するために TSP に送信するハッシュ値は、CompleteCertificateRefs 要素および CompleteRevocationRefs 要素を元に計算する。

2.3.5.1. SigAndRefsTimeStamp 要素

この要素に含まれるタイムスタンプトークンの計算は、Implicit Mode で実行される。具体的には、ds:SignatureValue 要素およびすべての SignatureTimeStamp 要素を正規化して連結する。次に、CompleteCertificateRefs 要素、CompleteRevocationRefs 要素をそれぞれ正規化し、XAdES 文書内の出現順序に従って連結する。連結した結果が TSA に送付されるダイジェスト値の計算の入力となる。リスト 2.3.5.1.1 に SigAndRefsTimeStamp 要素の XMLSchema 定義を示す。

リスト 2.3.5.1.1：SigAndRefsTimeStamp 要素の XMLSchema 定義

```
<xsd:element name="SigAndRefsTimeStamp" type="XAdESTimeStampType"/>
```

2.3.5.2. RefsOnlyTimeStamp 要素

この要素には、CompleteCertificateRefs 要素、CompleteRevocationRefs 要素を連結したデータを下に計算したタイムスタンプが格納される。この要素に含まれるタイムスタンプトークンの計算は、Implicit Mode で実行される。具体的には、CompleteCertificateRefs 要素、CompleteRevocationRefs 要素をそれぞれ正規化し、XAdES 文書内の出現順序に従って連結する。連結した結果が TSA に送付されるダイジェスト値の計算の入力となる。リスト 2.3.5.2.1 に RefsOnlyTimeStamp 要素の XMLSchema 定義を示す。

リスト 2.3.5.2.1：RefsOnlyTimeStamp 要素の XMLSchema 定義

```
<xsd:element name="RefsOnlyTimeStamp" type="XAdESTimeStampType"/>
```

2.3.6. Extended long electronic signatures with time (XAdES-X-L)

XAdES-X-L は、XAdES-X type1 か type2 のいずれかに対して、UnsingeSignatureProperties 要素に CertificateValues 要素と RevocationValues 要素を加えたものである（UnsignedSignatureProperties 要素の XMLSchema 定義はリスト 2.2.5.1 を参照のこと）。

なお、後述する XAdES-A 形式を構成する場合、XAdES-X type1 や type2 に対して

XAdES-X-L を構成する必要はなく、CeritificateValues 要素と RevocationValues 要素があれば良い。

2.3.6.1. CertificateValues 要素

この要素は、署名者証明書および CompleteCertificateRefs 要素で参照される証明書チェーンを含まなければならない(MUST[22]7.6.1)。ただし、ds:Signature 要素内の ds:KeyInfo 要素に既に含まれている証明書は CertificateValues 要素内に含む必要はない。リスト 2.3.6.1.1 に CertifiCateValues 要素の XMLSchema 定義を示す。

リスト 2.3.6.1.1：CertificateValues 要素の XMLSchema 定義

```
<xsd:element name="CertificateValues" type="CertificateValuesType"/>

<xsd:complexType name="CertificateValuesType">
    <xsd:choice minOccurs="0" maxOccurs="unbounded">
        <xsd:element name="EncapsulatedX509Certificate"
                     type="EncapsulatedPKIDataType"/>
        <xsd:element name="OtherCertificate" type="AnyType"/>
    </xsd:choice>
    <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>
```

リスト 2.3.6.1.2 に CertificateValues 要素の例を示す。

リスト 2.3.6.1.2：CertificateValues 要素の例

```
<xa:CertificateValues Id="CertVal-ID">
  <xa:EncapsulatedX509Certificate>Xt1HwwQjs.....iUrU5itXe+X=</xa:EncapsulatedX509Certificate>
</xa:CertificateValues>
```

2.3.6.2. RevocationValues 要素

この要素には、署名検証に必要な証明書の証拠情報が格納される。この要素は一つの XAdES 署名文書について一つしか存在しない。リスト 2.3.6.2.1 に RevocationValues 要素の XMLSchema 定義を示す。

リスト 2.3.6.2.1：RevocationValues 要素の XMLSchema 定義

```
<xsd:element name="RevocationValues" type="RevocationValuesType"/>

<xsd:complexType name="RevocationValuesType">
  <xsd:sequence>
    <xsd:element name="CRLValues" type="CRLValuesType" minOccurs="0"/>
    <xsd:element name="OCSPValues" type="OCSPValuesType" minOccurs="0"/>
    <xsd:element name="OtherValues" type="OtherCertStatusValuesType"
                 minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID"  use="optional"/>
</xsd:complexType>

<xsd:complexType name="CRLValuesType">
  <xsd:sequence>
     <xsd:element name="EncapsulatedCRLValue"
                  type="EncapsulatedPKIDataType"
                  maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>


<xsd:complexType name="OCSPValuesType">
  <xsd:sequence>
      <xsd:element name="EncapsulatedOCSPValue"
                   type="EncapsulatedPKIDataType" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

リスト 2.3.6.2.2 に RevocationValues 要素の例を示す。

リスト 2.3.6.2.2：RevocationValues 要素の例

```
<xa:RevocationValues Id="RevoVal-ID">
  <xa:CRLValues>
    <xa:EncapsulatedCRLValue>MIICFAdHARJMEe ..... OrFUp5t+xug==</xa:EncapsulatedCRLValue>
  </xa:CRLValues>
</xa:RevocationValues>
```

ETSI TS 101 903 V1.3.2(2006-03)[22]では、署名の中に属性証明書が含まれる場合は、AttributeCertifcateValues 要素と AttributeRevocationValues 要素も加えることが出来るように定義されている。しかし、本書の効果的かつ効率的（必要最低限の）プロファイルを提示するという方針に従い、AttributeCertificate には言及しないこととする。

2.3.7. Archival electronic signatures (XAdES-A)

電子署名の長期保存のためには、署名対象文書、デジタル署名、タイムスタンプおよび

検証情報全体をタイムスタンプや耐タンパな仕組みで保護する必要がある。長期署名フォーマットでは、アーカイブタイムスタンプによりこれを実現する。このとき、XAdES-C や XAdES-X で利用した検証情報へのリファレンスや検証情報のリファレンスへのタイムスタンプは必要なくなり、検証情報そのものを CertificateValues や RevocationValues で確保し、アーカイブタイムスタンプを付与すればよい。

また、電子署名の検証可能期間を極めて長くしようとしたとき、タイムスタンプの署名の危殆化や TSA 証明書の有効期限切れが発生しうるため、タイムスタンプの署名を複数回重ねることが要求されることがある。このとき、ArchiveTimeStamp 要素が用いられる。このタイムスタンプは期間を置いて繰り返される。

2.3.7.1. ArchiveTimeStamp 要素

この要素は、アーカイブタイムスタンプが格納される。この要素に含まれるタイムスタンプのハッシュ値の計算は、タイムスタンプ対象となる全ての非署名属性と ArchiveTimeStamp プロパティ自体が同じ親を持っているかどうかで異なる。以下にその 2 つのケースについて説明する。

Not distributed case（分離されていないケース）

ArchiveTimeStamp とそのタイムスタンプ対象となるすべての非署名属性が同じ親を持つとき、Implicit メカニズムを利用する。Not distributed case の構造は図 2.3.7.1.1 のようになる。

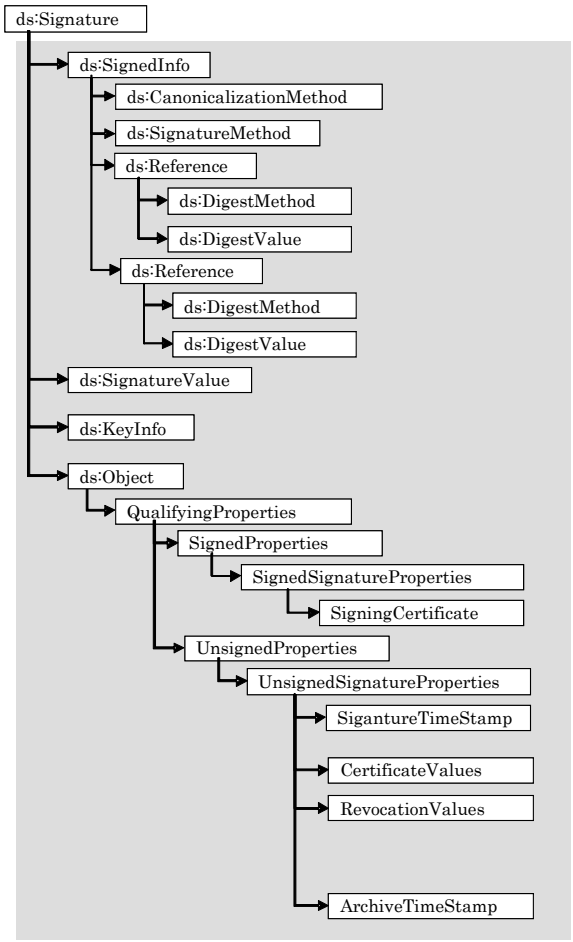


図 2.3.7.1.1 Not distributed case の構造

ダイジェスト値の計算の入力は、以下のように構築しなければならない(MUST[22]7.7.1)。

1. octet stream を初期化する。
2. ds:Reference 要素を取得する。
   - XML 署名の参照処理モデルによって、検索された ds:Reference 要素を処理する。
   - ds：Canonicalization が存在するならば、この要素で示されるアルゴリズムが使われる。
   - 処理した octet stream を連結する。
3. 出現する順に以下の要素をとって、それぞれを正規化して、処理した octet stream を連結する：
   - ds:SignedInfo 要素
   - ds:SignatureValue 要素
   - 存在した場合 ds:KeyInfo 要素
4. UnsignedSignatureProperties 内で現れる順に、以下の要素を取り出す

- 存在する SignatureTimeStamp 要素
- 存在する CounterSignature 要素
- 存在するならば CompleteCertificateRefs と CompleteRevocationRefs 要素
- 存在するならば AttributeCertificateRefs と AttributeRevocationRefs 要素
- 存在する SigAndRefsTimeStamp 要素と RefsOnlyTimeStamp 要素
- CertificateValues 要素。存在しないならば、この要素は加えられなければならない(MUST[22]7.7.1).
- RevocationValues 要素。存在しないならば、この要素は加えられなければならない(MUST[22]7.7.1).
- The AttrAuthoritiesCertValues 要素。存在せず、そして署名における属性証明書が存在し、その検証において使われた証明書が、CertificateValues に現れないならば、この要素は加えられなければならない(MUST[22]7.7.1).
- The AttributeRevocationValues 要素。存在せず、そして署名における属性証明書が存在し、その検証において使われたいくつかの証明書が、RevocationValues に現れないならば、この要素は加えられなければならない(MUST[22]7.7.1).
- 以前に存在する ArchiveTimestamp 要素

5. ds:Reference によって参照されていないすべての ds:Object を取得する。それぞれを正規化し、最終的な octet stream に各々の結果として生じる octet stream を連結する。ds：Canonicalization が存在するならば、この要素で示されるアルゴリズムが使われる。存在しないならば、XML 署名によって指定される標準的な正規化が使われる

Distributed case（分離されているケース）

ArchiveTimeStamp とそのタイムスタンプ対象となる非署名属性のいくつかが同じ親を持たないとき、explicit メカニズムを使わなければならない(MUST[22]7.7.2)。Distributed case の構造は図 2.3.7.1.2 のようになる。

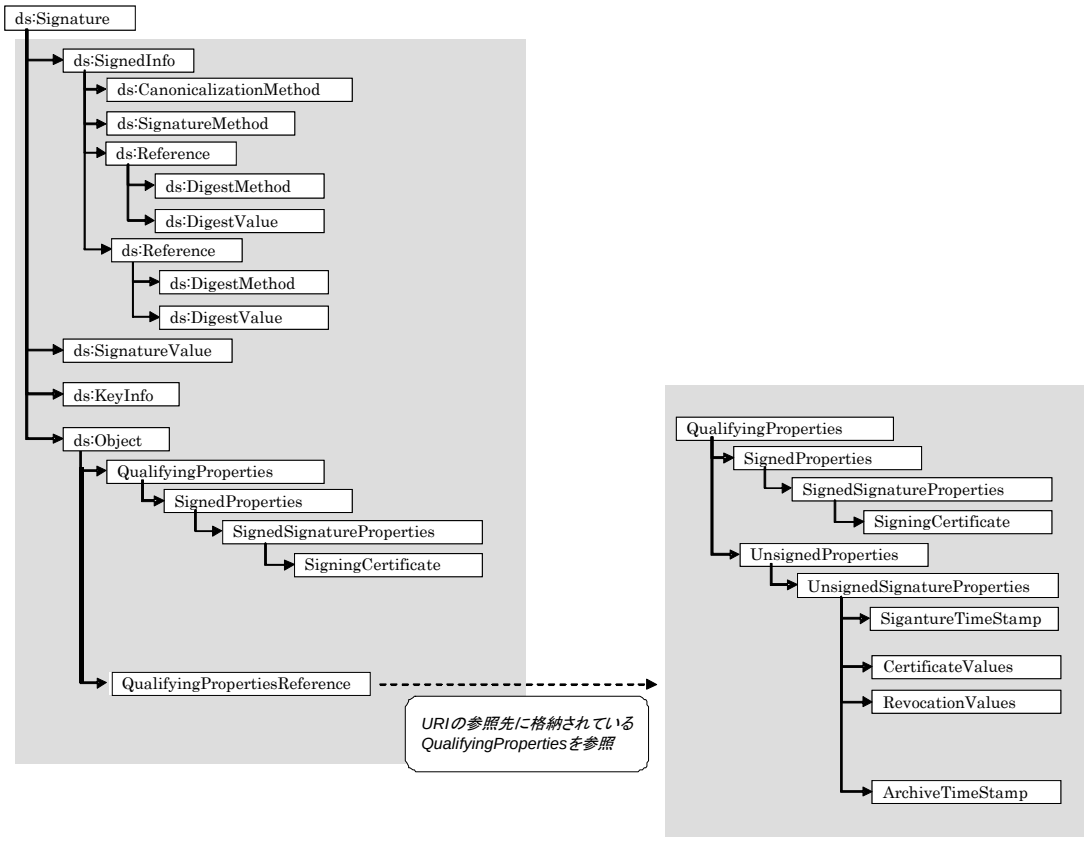


図 2.3.7.1.2 Distributed case の構造

ダイジェスト値の計算の入力は、以下のように構築しなければならない。

以下の unsigned signature properties を取得する。コメントノードを抜き出し、それぞれを正規化して、各々の結果として生じる octet stream を連結する。それらが ArchiveTimeStamp 要素で Include 要素に参照される順は、非署名属性は処理される順番と同一でなければならない(MUST[22]7.7.2)。

- 存在する SignatureTimeStamp 要素
- 存在する CounterSignature 要素
- 存在するならば CompleteCertificateRefs 要素
- 存在するならば CompleteRevocationRefs 要素
- 存在するならば AttributeCertificateRefs 要素
- 存在するならば AttributeRevocationRefs 要素
- 存在する SigAndRefsTimeStamp 要素
- 存在する RefsOnlyTimeStamp 要素

- CertificateValues 要素。存在しないならばこの要素を追加しなければならない(MUST[22]7.7.2).
- RevocationValues 要素。存在しないならばこの要素を追加しなければならない(MUST[22]7.7.2).
- AttrAuthoritiesCertValues 要素。存在しないならばこの要素を追加しなければならない(MUST[22]7.7.2).
- AttributeRevocationValues 要素。存在しないならばこの要素を追加しなければならない(MUST[22]7.7.2).
- 以前に存在する ArchiveTimestamp 要素

ダイジェストの計算の入力は以下のように構築する

1) 空の octet stream として最終的な octet stream を初期化する。
2) 署名者が SignedProperties 要素を含めて参照している ds:SignedInfo の中で、それらの登場順で全ての ds:Reference 要素を取得する。以下で示すようにそれぞれを処理する
    - XML 署名の参照処理モデルによって、検索された Reference 要素を処理する。
    - 結果が XML ノードセットならばそれを正規化する。ds：Canonicalization が存在するならば、この要素で示されるアルゴリズムが使われる。存在しないなら XML 署名によって指定される標準的な正規化方法が使われる。
    - 結果として生じる octet stream を連結する。
3) 以下の XML 署名要素をとって、それぞれを正規化して、最終的な octet stream に各々の結果として生じる octet stream を連結する。
    - ds:SignedInfo 要素
    - ds:SignatureValue 要素
    - 存在するならば ds:KeyInfo 要素
4) Include 要素の出現順で処理する。最終的な octet stream に結果として生じる octet stream を連結する。
5) ds:Reference によって参照されていないすべての ds:Object を取得する。それぞれを正規化して、octet stream を連結する。ds：Canonicalization が存在するならば、この要素で示されるアルゴリズムが使われる。存在しないなら、XML 署名によって指定される標準的な正規化方法が使われる。

リスト 2.3.7.1.1 に ArchiveTimeStamp 要素の XMLScheme 定義を示す。同時に複数の TSA へタイムスタンプのリクエストを送信した場合、得られる複数のタイムスタンプトー

クンは、一つの ArchiveTimeStamp 要素内に格納しなければならない。アーカイブタイムスタンプ自身の証明書検証情報（タイムスタンプの証明書からそのルート CA に至るまでの証明書パスおよびそれぞれの証明書の失効情報）の格納方法について、大きく分けて以下の 2 通りが考えられる。

① タイムスタンプトークンに TSA 証明書検証情報を埋め込む方法

② TSA 証明書の検証情報を別途保管する方法

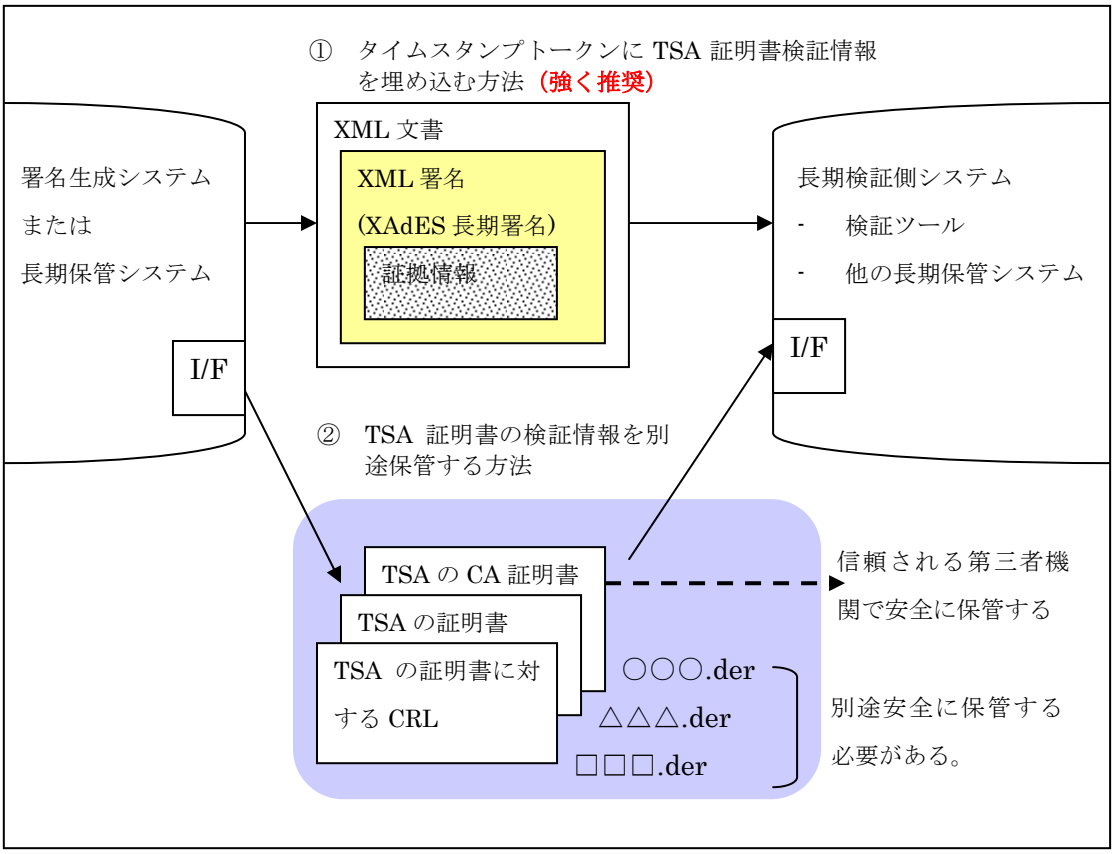


図 2.3.7.1.3 タイムスタンプトークンの証拠情報の扱いについて

タイムスタンプトークンの証拠情報の扱い方については①タイムスタンプトークン自身に埋め込む方法と②別途安全に保管する方法の二通りが考えられるが、本書では①を強く推奨する。

相互互換性の観点から考えると、検証方法を明確に定義できるのでプロファイルとしては①の方法を強く推奨する。また、①の方法を選択した場合の証拠情報の長期署名フォーマットへの格納方法は次の 2 通り（両方とも標準仕様では言及されていない）考えられるが、検証時には 1),2)を処理できることを推奨する。

1) タイムスタンプトークンの certificates と crls フィールド

2) タイムスタンプトークン内の unsigned attribute(Extended validation data 形式)

一方、②の方法は相互互換性の観点から考えると、別途保管したタイムスタンプの証拠情報を用いて検証者がタイムスタンプを検証できる必要があるため、検証者の検証プログラムがタイスタンプの証拠情報を読み込む機能とそれら使った長期署名フォーマットの検証を実行できる必要がある。

リスト 2.3.7.1.1：ArchiveTimeStamp 要素の XMLSchema 定義

```
<xsd:element name="ArchiveTimeStamp" type="XAdESTimeStampType"/>
```

リスト 2.3.7.1.2 に ArchiveTimeStamp 要素の例を示す。

リスト 2.3.7.1.2：ArchiveTimeStamp 要素の例

```
<xa:ArchiveTimeStamp Id="ATS-ID">
  <xa:Include URI="#Reference0" referencedData="true" />
  <xa:Include URI="#Reference1" referencedData="true" />
  <xa:Include URI="#SignedPropRef0" referencedData="true" />
  <xa:EncapsulatedTimeStamp
  Encoding="http://uri.etsi.org/01903/v1.2.2#DER">73/gEwDQYJ1a1knynA==</xa:EncapsulatedTimeStamp>
</xa:ArchiveTimeStamp>
```

第一部 デジタル署名ハンドブック

II. デジタル署名の生成

本節では長期署名フォーマット(ES,ES-T,ES-C,ES-XL,ES-A)の生成方法について述べる。最初に CAdES、XAdES に共通する生成の基本事項、留意事項について述べ、次に CAdES、XAdES それぞれの生成プロセスについて解説する。

1. CAdES、XAdES に共通の事項

1.1. 生成の概要

これまで述べてきたように長期署名フォーマットには、データ構造を持つバイナリ形式である ASN.1 BER 形式に基づく暗号メッセージ構文(CMS: Cryptographic Message Syntax)に基づいた CAdES 形式と、XML 署名に基づいた XAdES 形式がある。

CAdES および XAdES では、非署名属性群(unsignedAttrs)もしくは非署名プロパティ群(UnsignedProperties)と呼ばれる、署名者の署名対象とはならない領域をそれぞれ持っており、ここに、長期保存を可能にするための情報として以下の情報を格納することにより、署名対象の長期的な保存を可能にしている。

・署名タイムスタンプ	署名時刻を第三者的に確定するためのタイムスタンプ
・検証情報	署名の検証に必要な証明書および失効情報
・アーカイブタイムスタンプ	上記 2 つを含む署名データを保護するタイムスタンプ

従って、長期署名フォーマットを生成するためには、最初に既存の CMS SignedData 型の署名データや XML 署名を生成するプログラムにより一般的な署名データを生成し、そして単に上述の非署名属性/プロパティを追加していくというのが、基本的な処理内容である。

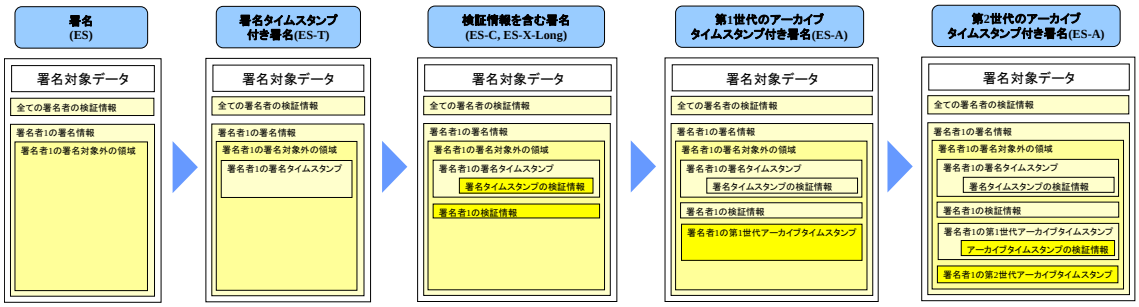


図 1.1.1: 長期署名フォーマットの生成

1.2. 各フォーマットの生成者について

署名データおよび長期署名データを生成するために、必ずしも個人または機器である署名生成プログラムが最初の生成から、タイムスタンプ、検証情報の付与まで、全てを行わなければならないというわけではない。最初の署名だけ、署名を行いたい本人が行う必要

があるが、以降の役割は本人が行うことも、サーバー側で行うことも可能である。基本的な考え方としては、証を立てたい当人が正しいデータを作成するというのが設計ポリシになるだろう。

1.3. CAdES/XAdES-C/X のための失効検証情報取得のタイミングについて

署名の長期保存のためには、署名された時点で使われた証明書が有効であったかを数年後といった時間単位で、後に検証できるように検証に必要な全ての証明書や失効情報をアーカイブに含める。

証明書は取得時刻に依存するところはあまり無いが、失効情報に関しては、アーカイブに含める検証に必要な正当な失効情報を取得するタイミングに配慮する必要がある。失効情報の取得において、許されるタイミングは証明書ポリシ(CP)や認証実施規定(CPS)で規定された認証局サービスに依存するし、署名者や検証者、その利用するソフトウェアの要件にも依存する。

本節では、アーカイブに含める失効情報を取得する時刻が、何故重要な問題であるのかを解説し、失効情報を取得すべきタイミングのガイドラインを示す。

【問題 1】署名時刻(or 署名タイムスタンプ時刻)に取得した失効情報ではだめなのか

署名者が自身の証明書の失効申請をしてから CRL に失効情報が記載されるまで、認証局の認証実施規定(CPS)にも依存するが、受理次第即座に反映されるものから、24 時間から数営業日を要する場合もあるかもしれない。従って署名の長期保存アーカイブを作成する場合に含める失効情報は、署名時刻における失効状態が正しく判定できるように、失効の有無が反映されるまでの時間を待って失効情報を取得しなければならない。署名時刻と同時に取得した失効情報をアーカイブに含めるのは、一般的には検証情報として不適切である。

【問題 2】理想的な CRL の取得時刻と現実のシステム

では、証明書の失効状態を知るために理想的な CRL 取得時刻はいつだろうか。これは、証明書の有効期間の終わりの直前であれば、失効されているとすれば、必ず CRL に記載されているはずなので、理想論からすれば、この時刻で取得するのがよいが、通常ではこの待つ時間は署名者証明書ならば 1 年～2 年、タイムスタンプ局証明書ならば 10 年といった期間となり、この時刻に至るまで失効情報の取得ならびにアーカイビングを待つシステムを実装するのは現実的ではないだろう。そのため、後で説明する猶予期間(Grace Period)を考慮することとなる。

【問題 3】証明書有効期間の後でも CRL に失効情報が記載されるかどうか

認証局によって、CRL への失効情報の記載が証明書の有効期間のみ CRL に記載する場合と、証明書の有効期間後も一定期間、もしくはずっと記載し続ける場合がある。検証用の CRL を自動的に取得するシステムの場合には、このような認証局の違いにも留意する必要がある。

【問題 4】猶予期間(Grace Period)とは

署名時刻に取得した失効情報では検証するために十分ではなく、証明書の有効期限の終了まで長すぎてアーカイビングを待てないため、猶予期間(Grace Period)という考え方を導入する。猶予期間とは、署名時刻と失効時刻が近かった場合でも、署名者証明書の有効性を正しく判断できるように、署名時刻付近の失効情報の反映を待ってから失効情報を取得し、これをアーカイブ対象の検証情報とする待ち時間のことである。失効情報を発行する認証局の CPS にも依存するが、一般的には翌日から数営業日後となる。

猶予期間については署名者と検証者で合意を取る必要がある。

CRL には失効情報の取得時刻は記載されていないため、thisUpdate で猶予期間を満足しているか判断することとなる。CRL の発行周期が長く証明書検証時刻が thisUpdate より後の場合には、この CRL が検証に有効なものであるかどうかは、CRL の更新履歴など自動処理できない別の方法で判断することとなる。

【問題 5】thisUpdate, nextUpdate, CRLNumber や緊急失効について

注意しなければならないのは、証明書失効リスト(CRL)は証明書と異なり、有効期間という考え方が無く、thisUpdate から nextUpdate に記された期間、発行された CRL は有効であるということにはならない。認証局によっては、失効申請があった場合、次の定期的な CRL 発行周期まで待たずに即座に CRL を発行するような緊急失効や即時失効のサービスを提供しているものがあるかもしれない。

thisUpdate は CRL の発行された時刻であるが、nextUpdate については、その時刻まで CRL が有効なのではなく、その時刻までに緊急失効が無かったとしても、その時刻までには必ず次の CRL を発行するという宣言である。RFC 3280[6]においては nextUpdate はオプションであり、必ずあるとは限らない。CRL プロファイルに nextUpdate が無いような認証局の場合には、証明書が失効されない限りは CRL も更新されないだろう。

最新の CRL が何であるか、CRL の発行の順序関係を調べるためには CRL の CRLNumber 拡張を利用することができる。CRLNumber は連続番号とは限らないが、必ず時間的に前の番号よりも大きくなるようになっているので、ある時点での証明書有効性を検証するために、どれが最新の CRL かは thisUpdate, nextUpdate, CRLNumber に記載された情報を元に判断することとなる。

【問題 6】CRL の発行周期について

CRL の定期発行周期が非常に長いか、定期には CRL を発行せず、失効申請があり次第 CRL を発行するような認証局を利用する場合、アーカイブに含める CRL において、署名時刻よりも thisUpdate が前であるということは有り得る。この場合には、CRL が該当する証明書を検証するのに正当なものであるかどうかは自動的に判定することはできず、CRL の発行履歴と照らし合わせて正当性を確認することとなる。

【問題 7】OCSP ならばリアルタイムに失効状態がわかるのか

オンライン証明書状態プロトコル OCSP は RFC 2560 [1]で規定された一つまたは複数の証明書の現在の失効状態を有効/無効の形式で取得するためのプロトコルである。証明書の発行枚数が多い認証局である場合には、一般に証明書失効リスト(CRL)も大きくなり、キャッシュなどの機能を使わない場合にネットワークトラフィックを圧迫する。このような場合にある特定の証明書の失効状態を知ることができるようにした軽量なプロトコルである。OCSP を使えば、署名者証明書の有効性をリアルタイムで知ることができるように思われるかもしれないが、必ずしもそうとは限らない。

認証局や OCSP レスポンダのシステムは一般的に、ユーザの失効状態をデータベースで管理し、CRL を発行し、その CRL の状態に基づいて OCSP 応答を返しているものがほとんどである。この場合には、証明書の失効状態の新鮮さという意味において、CRL も OCSP 応答も全く変わらないこととなる。また、失効情報の取得方法によって有効/無効が違うという問題もあるため CRL でも OCSP 応答も失効結果は同じくなるようにしていると思われる。

従って、一般的に OCSP を使えば検証時刻と同時刻にアーカイブ用の失効情報を取得できることにはならない。

【まとめ】アーカイビングのための失効情報取得時刻のまとめ

以下に列挙する観点から、署名時刻後、いつ取得した失効情報ならば正当な失効情報であると認めるか、署名者と検証者の間で合意形成をする必要がある。

・署名者証明書とタイムスタンプ局証明書用の失効情報の発行周期
・並びに上記の証明書チェーンを構成するサブ CA 証明書の失効情報の発行周期
・失効申請から CRL に反映されるまでの時間(Grace Period)
・証明書有効期間の後でも CRL に失効情報が記載されるかどうか
・緊急失効(定期 CRL 発行以外の CRL 発行)の有無

一般的には、署名者証明書の失効情報は、署名してから 2～3 営業日後とし、署名タイムスタンプのタイムスタンプ局証明書についてもこれに合わせるかもしれない。

署名の長期保存により期間を経た後、これより短い時刻で証明書の有効性を判断する必要がある場合には、自動的に署名の有効性を判定することはできず、様々な状況証拠、失効情報、申請情報などを署名者、検証者、認証局、タイムスタンプ局などから集め法的な場での係争となるだろう。

また、将来的には日本国内では認証局に失効情報のセキュアなアーカイビングと公開を義務付ける動きもあり、このようなサービスを利用すれば、ある時点での失効検証のための CRL が公開情報として取得できようになるかもしれない。

1.4. CAdES と XAdES の属性/プロパティの違い

CAdES と XAdES は、対称性を持つように設計されており、ほぼ同じ署名および非署名の属性/プロパティが定義されているが、若干の違いもある。

表 1.4.1: CAdES v1.7.3 と XAdES v1.3.2 の署名および非署名の属性/プロパティの比較

CAdES	XAdES	用途
signedAttrs	SignedProperties	署名対象の属性/プロパティ群を格納する
MessageDigest	※Referenceに記載	署名対象文書のハッシュ値
ContentReference	-	別の署名データと依存関係がある場合の参照先ContentIdentifier
ContentIdentifier	-	この署名の識別子
-	SignedSignatureProperties	署名対象となる署名のプロパティ
SigningTime	SigningTime	署名者のローカルクロックによる保証のない署名時刻
ESSCertV1/V2/Other	SigningCertificate	署名者の証明書情報
SignaturePolicyIdentifier	SignaturePolicyIdentifier	署名ポリシ識別子
SignerLocation	SignatureProductionPlace	署名者の所在地、署名を生成した場所
SignerAttributes	SignerRole	署名者の認定された資格属性、認定されていない属性
-	SignedDataObjectProperties	署名対象となる署名対象データのプロパティ
ContentHints	DataObjectFormat	署名対象データのデータ形式、種類、MIMEタイプ
CommitmentTypeIndication	CommitmentTypeIndication	この署名の意図
ContentTimeStamp	AllDataObjectsTimeStamp	(全てのデータに対する)コンテントタイムスタンプ
-	IndividualDataObjectsTimeStamp	(個々のデータに対する)コンテントタイムスタンプ
unsignedAttrs	UnSignedProperties	署名対象とならない属性/プロパティ群を格納する
-	UnsignedSignatureDataProperties	署名対象とならない署名に関するプロパティ
CounterSignature	CounterSignature	カウンタ署名
SignatureTimeStamp	SignatureTimeStamp	署名タイムスタンプ
CompleteCertificateRefs	CompleteCertificateRefs	署名者証明書の証明書チェーンを構成する参照情報
CompleteRevocationRefs	ComplateRevocationRefs	署名者証明書の検証に必要となる全失効情報の参照情報
AttributeCertificateRefs	AttributeCertificateRefs	属性証明書の参照情報
AttributeRevocationRefs	AttributeRevocationRefs	属性証明書の失効情報の参照情報
ESCTimeStamp	SigAndRefsTimeStamp	署名と検証参照情報へのタイムスタンプ
TimestampedCertsCRLs	RefsOnlyTimeStamp	検証参照情報のみへのタイムスタンプ
CertificateValues	CertificateValues	署名者の認証パスを構成する証明書群
RevocationValues	RevocationValues	署名者の認証パスの失効検証に必要な失効情報群
-	AttrAuthoritiesCertValues	署名者の属性証明書群
-	AttributeRevocationValues	署名者の属性証明書の失効検証に必要な失効情報群
ArchiveTimeStamp(V1/V2)	ArchiveTimeStamp	アーカイブタイムスタンプ

上表より名称に若干の違いがあるのと、CAdES では属性証明書およびその失効情報を格納する場所が無いことがわかる。

CAdES と XAdES の双方を実装者が留意しなければならない点として、OCSP による失効情報が挙げられる。失効情報を格納する CompleteRevocationRefs、RevocationValues の要素において、CAdES では BasicOCSPResponse に基づき値を格納するが、XAdES では OCSPResponse に基づく値を格納する。データ構造は RFC 2560[1] 4.2.1 節で規定されており OCSP 結果ステータス、データ型、および BasicOCSPResponse をラップしたのが OCSPResponse である。

2. CAdES

2.1. CAdES フォーマットデータの生成の基本事項

本節では CAdES フォーマットデータの生成について述べる

生成処理は、以下の処理に分割される。

- 基本的な署名データの生成(ES-BES/ES-EPES)
- 署名タイムスタンプの付与(ES-T)
- 証明書検証情報の付与(ES-C, ES-X-Long)
- アーカイブタイムスタンプの初回付与(ES-A)
- 2 回目以降のアーカイブタイムスタンプの付与(ES-A)

2.1.1. CAdES-BES, CAdES-EPES の生成

基本的な署名データである CAdES-BES、またこれに署名ポリシ情報を追加した CAdES-EPES の生成について説明する。データ構造の生成については CMS 署名データ (CMS SignedData) [10] が基本となるが、幾つか必須となる署名属性がある点が若干異なる。

まず、CAdES-BES、CAdES-EPES のいずれについても、署名者を特定するための情報である SigningCertificate 属性が必須となるため、CMS ではオプションとなっている署名属性群フィールド(signedAttrs)が必須となる。

生成の処理手順は以下のようになる。

①署名属性群 signedAttrs の要素となる属性の生成

- MessageDigest 属性の生成
  SignerInfo の digestAlgorihm フィールドで指定されたアルゴリズムにより署名対象データのオクテット全体に対しハッシュ値を計算し、属性を生成する。
- SigningCertificate 属性の生成
  署名者証明書に対し発行者、シリアル番号、ハッシュ値などの情報を取得し、署名者証明書情報の属性を生成する。
- SignaturePolicyIdentifier 属性の生成(オプション：CAdES-EPES の場合のみ)
  署名ポリシ識別子の属性を生成する。

②署名属性群 signedAttrs の DER 正規化

署名属性群を表す[0] IMPLICIT SET OF 構造に全ての署名属性要素を設定し、適用されていなければ要素のソートなどの DER による正規化を行う。

③署名値の計算と設定

署名属性群 signedAttrs フィールドに対し、[0] IMPLICIT SET OF を SET OF に置き換え、これを署名対象とし signatureAlgorithm フィールドで指定された署名アルゴリズムにより署名値を計算し signature フィールドに値を設定する。

④他のフィールドの設定

2.1.2. CAdES-T の生成

先に生成した CAdES-BES、CAdES-EPES フォーマットに対して署名タイムスタンプ属性非署名属性群(unsignedAttrs)に追加することにより CAdES-T フォーマットとなる。具体的には SignerInfo 構造において署名値を表す signature フィールドの値オクテットに対して、タイムスタンプトークンを取得する。

生成の処理手順は以下のようになる。

①署名値オクテットの取得

SignerInfo の signature フィールドの値オクテットを取得する。

②署名値に対するタイムスタンプトークンの取得

- タイムスタンプサービスに接続するための初期設定
  ホスト名や IP アドレス、TCP ポート、有償サービスの認証情報(ID/パスワード等)を設定する。
- 署名値オクテットに対してハッシュ値を計算し、タイムスタンプ要求のデータ構造を生成する。
- タイムスタンプ要求をタイムスタンプサービスに送信する。
- タイムスタンプ応答を取得する。
- 応答が正しいか検証する。
  応答のステータス値を確認し、要求で送ったノンス値(乱数値)と同じであるか、要求したハッシュ値に対する応答であるか検証し、必要に応じタイムスタンプ局(TSA)証明書の認証パス検証を行う。
- 応答よりタイムスタンプトークンを取り出す。

③属性値に取得したタイムスタンプトークンを設定した署名タイムスタンプ属性を生成

④署名タイムスタンプ属性を非署名属性群(unsignedAttrs)に追加する。

タイムスタンプトークンの取得処理については、有償のタイムスタンプサービスを利用した場合、サービス提供者が提供するソフトウェア開発キット(SDK)を利用することとなるので、一般的には処理内容について気にする必要はないだろう。

2.1.3. CAdES-C、CAdES-X-Long の生成

CAdES-T フォーマットから CAdES-C、CAdES-X-Long フォーマットを生成するためには、署名者証明書、カウンタ署名の署名者証明書、コンテントタイムスタンプ並びに署名タイムスタンプの TSA 証明書の検証に必要な情報を格納することにより行う。

CAdES-C は、証明書や失効情報のハッシュ値などを参照情報として持たせるために用いられ、CAdES-X-Long では証明書、CRL、OCSP レスポンスそのものを格納する。

CAdES-C と CAdES-X-Long は、別々のステップで生成することは同じ処理の繰り返しになり非効率であるため、CAdES-C から CAdES-X-Long を生成するということはなく、CAdES-T から直接それぞれを生成することになるだろう。

生成の処理手順は以下のようになる。

①全ての証明書の認証パスの構築と検証

署名者、TSA など検証情報を格納する必要のある全ての証明書に対して、これら証明書から信頼するルート証明書(トラストアンカ)までの認証パスを構築し、検証を行ない、証明書のチェーンとしてこれを収集する。また、証明書の失効検証の際に使用した CRL や OCSP レスポンスもチェーンを構成する順序で収集する。

②検証参照情報のための属性の生成(オプション：CAdES-C, CAdES-X-Long の場合)

証明書や失効情報の参照情報を格納する必要がある場合、CompleteCertificateRefs, CompleteRevocationRefs 非署名属性にハッシュ値、発行元、シリアル番号などの情報を格納する。

③検証情報の属性の生成(オプション：CAdES-X-Long の場合)

証明書、CRL、OCSP レスポンスそのものを格納する場合 CertificateValues, RevocationValues 非署名属性に格納することができる。

④検証情報の格納フィールド(オプション)

SignerInfo の上位となる CMS SignedData の certificates, crls フィールドに証明書や失効情報を格納することができる。但し、ここに格納する場合、全ての署名者で一つのフィールドとなるので、生成側、検証側は注意する必要がある。

格納する CRL と OCSP レスポンスについては、前述の「CAdES/XAdES-C/X のための失効検証情報取得の猶予期間について」でも述べたように、猶予期間を留意した取得が必要である。

2.1.4. 第一世代 CAdES-A の生成

署名対象文書、署名情報、証明書検証情報など全ての情報を暗号アルゴリズムの危殆化から保護するためにアーカイブタイムスタンプを追加することにより CAdES-A フォーマットを生成する。

アーカイブタイムスタンプを付与する前には、格納すべき全ての証明書検証情報が格納されており、これ以降はアーカイブタイムスタンプ以外の属性は追加しない状態になっていなければならない。

アーカイブタイムスタンプのハッシュ対象など生成に関する詳細はフォーマット詳細の節をご覧頂きたい。

2.1.5. 第二世代以降の CAdES-A の生成

過去の CAdES-A フォーマットを、暗号アルゴリズムの危殆化および直前のアーカイブタイムスタンプの TSA 証明書の期限切れから保護するために、追加のアーカイブスタンプを付与することにより第二世代以降の CAdES-A フォーマットが生成される。

アーカイブタイムスタンプを追加する前に、直前のアーカイブタイムスタンプの TSA 証明書の検証情報を、そのタイムスタンプトークン内に格納した上で、アーカイブタイムスタンプ属性を追加する。

2.2. CAdES フォーマットデータの生成における注意事項

本節では、CAdES のコアライブラリの実装者が留意しなければならない点について述べる。生成と検証を別のシステムが行う場合や、署名データの生成と署名タイムスタンプの追加、検証情報の追加、アーカイブタイムスタンプの追加を、それぞれ別のシステムが行う場合には本節で解説された内容について確認する必要がある。相互運用性に問題が生じた場合においても、本節を確認するのがよい。

2.2.1. ASN.1 BER と DER の違い

ASN.1 には幾つかのエンコーディング方法が規定されており、PKI やセキュリティの分野で多用されているのが DER と BER である。以下に一般的な利用例をまとめた。

DER: X.509 証明書、CRL、RFC 3161 タイムスタンプ
BER: S/MIME, CMS, CAdES

BER と DER のエンコーディング方法の包含関係は以下のようであり、DER データであ

れば、これは BER であるともいえる。DER のデータ表現は必ず同じ意味を持つデータ内容を一意に表現できる方法となっているが、BER においては、そのデータのエンコーディング方法は幾つかの選択肢がある。

一般に ASN.1 ではタグと呼ばれるデータ種別、値データの長さ、および値をバイトで表すデータ構造となっている。

長期署名フォーマットは BER に基づいているということを生成および検証プログラムの実装者は留意する必要がある。長期署名フォーマットにおいて留意しておくべき、BER と DER の違いをまとめたのが以下である。

- SET OF 構造において、DER はデータ長順でソートされるが、BER ではソートされない。
- BER は DER と異なり不定長データによる表現も可能である。

これまで、ETSI および ECOM における実証実験および調整から得られた基本的な合意として、「生成側は入力に対して、既存の部分に対して不要な変更を与えずそのままの形で出力する。」そして「検証側はそのままの形で検証をする」としている。即ち、生成システムの入力が以下のことをしてはならないことを意味している。

- 生成の際、入力として読み込んだデータの SET OF 構造の順序を変えず、その順序で出力する。SET OF に追加する場合は、最後に追加する。本事項は長期署名において追加変更される UnsignedAttributes において特に注意する。また、他の SET OF 構造を持つデータについて変更をしないように注意する。
- 既存の BER 不定長表現を DER 固定長表現にするような変更をしてはならない。

2.2.2. アーカイブタイムスタンプのバージョン非互換性について

CAdES のアーカイブタイムスタンプのハッシュ計算法にはバージョンの違いにより以下の 3 通りの計算方法があることを「フォーマット詳細」において述べた。ETSI TS 101 733 のバージョンの差異についてまとめたのが以下となる。

表 2.2.2.1: ETSI TS 101 733 CAdES のバージョンの差異

ETSI CAdES	～1.4.0('02 09)	1.5.1('03 12)	1.6.3('05 09)	1.7.3('07 02?)
ETSI TS 101 733 CAdES 改定の特徴	・ArchiveTimeStamp (AT)の曖昧性の少ない計算法定義 ・CounterSignaturesはAT保護対象外	・AT計算方法の大幅変更 ・ATハッシュ計算方法の規定が曖昧 ・CounterSignatures など他の非署名属性もATで一括保護 ・署名ポリシのないBES	・CAdESとの呼称変更 ・XAdESとの一貫性 ・EPES sigPolicyHash OPTIONAL	・ATハッシュ計算法を若干改定・曖昧性排除 ・ATのOID変更 ・外部署名をATで保護 ・ESSCertV2 ・CMS v3制限の削除
署名者証明書参照のハッシュ危殆化対策	OtherSigningCertificate 1.2.840.113549.1.9.16.2.**19**		MD5の排除	ESSCertV2 1.2.840.113549.1.9.16.2.**47**
ATの特徴	・ASN.1 TLV構造のTLの除去処理 ・AT対象は非署名属性個別要素単位	・ASN.1 TLV構造のTLを除去しない ・signerInfoの各要素をAT対象 ・unsignedAttrsはまとめて対象に変更 ・特に署名延長時のATの生成検証処理が不明瞭 ・外部署名の処理が不明瞭 ・互換性のないATが同じOID		・不明瞭なハッシュ対象の計算が明確化された。基本はAS IS ・外部署名の処理の明示 ・ATが新OID
AT OID	1.2.840.113549.1.9.16.2.**27** (v1)	ATの大幅変更		1.2.840.113549.1.9.16.2.**48** (v2)
RFC CAdES	3126('01 09) bassed on RFC 2630 CMS and ETSI TS 101 733 v1.2.2			3126の後継 on 3852
RFC CMS	2630('99 06)	3369('02 08)	3852('04 06)	
日本のガイドライン	ECOM 2005 Profile('05 06)			JIS('07 04?)
日本のガイドラインのETSIとの差異	・v1.5.1に基づくがATのみ～v1.4.0orRFC3126 ・相互運用性確保のための解釈の追加補足 ・外部署名でもAT保護対象 ・フォーマット要件(BES,EPES,T,C,XL,A) ・必須領域の定義 ・タイムスタンプ検証情報の格納法の規定			・最新に基づく(今はv1.7.3) ・必須領域の定義 ・フォーマット要件(BES,T,A) ・ECOM 2005 Profileベースも OK(一部注意が必要)

今後実装するのであれば、相互運用性の課題が多くの面で解決された V1.7.3 ベースで実装することを推奨する。

2.2.3. ETSI TS 101 733 CAdES v1.7.3 の注意すべき変更点

2007 年初旬公開される予定の ETSI TS 101 733 CAdES フォーマットの最新版である v1.7.3 では、これまでの ETSI/ESI TC と ECOM との協調関係により、数多くの ECOM の提言が盛り込まれた形になっており、相互運用性上問題となる不明瞭であった点が解消されている。CAdES に関する JIS 標準では、最新のものに従うとしており、JIS が公開される時点で最新となる v1.7.3 に基づくものとなるため、これまで旧仕様に基づいていた実装をしていた者は、その差分について十分な理解が必要である。CAdES v1.7.3 の公開に従い、ほぼ同じ内容で IETF RFC 3126 の改訂版が公開される予定である。

CAdES の JIS で特に注意しておかなければならない v1.6.3 からの変更点をまとめたのが以下である。

- ArchiveTimeStamp の方式の変更
  v1.6.3 から、さらに若干の変更が加えられた。不明瞭とされる点が明確化され、分

離署名のための処理が追加されている。詳細は「フォーマット詳細 1.5CAdES-A」で解説している。

- ESSCertV2 の利用と OtherSigningCert の非推奨
  近年報告されている SHA1 ハッシュ危殆化の問題に対応するため、署名者証明書参照情報 SigningCertificate 属性において他の強固なアルゴリズムが利用できるように OtherSigningCert 属性が提供されていたが、IETF S/MIME ワーキンググループにおいて ESSCertV2 属性がインターネットドラフトとなったことから、これを利用する。
- SignedData v3 に限定しなくなった。
  CAdES ではバージョン 3 の CMS SignedData でなければならないとされていたが、この制限により署名者属性証明書を含めるか、署名者の SignerInfo のバージョンが 3、即ち IssuerSerial ではなく署名者証明書の SubjectKeyIdentifier の値を入れなければならなかった。v1.7.3 では、そのような制限がないため、任意の証明書を利用した IssuerSerial による SignedData v1 でも構わなくなった。従って、構成要素に注意すれば PKCS #7 による BES もしくは EPES フォーマットに基づく長期署名フォーマットも可能となった。

3. XAdES

3.1. XAdES フォーマットデータの生成の基本事項

本節では XAdES フォーマットデータの生成について述べる。

生成処理は、以下の処理に分割される。

- 基本的な署名データの生成(XAdES-BES/XAdES-EPES)
- 署名タイムスタンプの付与(XAdES-T)
- 証明書検証情報の付与(XAdES-C, XAdES-X-Long)
- アーカイブタイムスタンプの初回付与(XAdES-A)
- 2 回目以降のアーカイブタイムスタンプの付与(XAdES-A)

3.1.1. XAdES-BES の生成

3.1.1.1. XAdES-BES 形式署名データの構築

detached 形式の XAdES-BES を例に XAdES-BES の生成手順を示す。XML 署名データを構築する際、以下の点に留意して手順で XAdES-BES 署名データを構築する。

・ 署名対象データを ds:Reference 要素の URI 属性で参照する。
・ SigningCertificate 要素とそれに必要な要素（SignedProperties 要素などの上位要素）を生成し、署名者証明書への参照情報を SigningCertificate 要素に格納する。
・ SignedProperties 要素を ds:Reference 要素で参照することで署名対象に追加する。

※. XAdES-EPES 形式を生成したい場合は、SigningPolicyIdentifier 要素を生成し、署名ポリシの識別子を格納する必要がある。

3.1.1.2. XAdES-BES 形式署名データの生成

構築された XML 署名データに対する署名値を以下の方法で計算する（括弧内の数字は図 3.1.1.2.1 の数字に対応する）。

①. ds:Reference 要素で参照されている署名対象文書を取得する。
②. ds:Referenec 要素の配下に変換処理（ds:Transform）が指定されていれば、取得した署名対象文書を変換する。
③. ds:DigestMethod で指定された方法で、その変換結果のハッシュ値を計算する。
④. 計算したハッシュ値は base64 にエンコードし ds:DigestValue 要素に格納する。
⑤. XAdES-BES は、ds:Reference 要素で SingedProperties 要素を参照している。従って、SignedProperties 要素とその配下の要素を取り出す。
⑥. このとき、SignedProperties 要素は同一の XML 文書内のノードなので ds:Transform 要素の指定が無くても正規化が実行される（node-set から octet stream への変換）

⑦. 同様にハッシュ値を計算する。
⑧. 計算したハッシュ値は base64 でエンコードし ds:DigestValue 要素に格納する。
⑨. SignedInfo 要素を取り出し、ds:CanonicalizationMethod 要素で指定する手法で正規化を行う。
⑩. それに対し SignatureMethod 要素で指定された署名処理を行う。
⑪. 署名値は、base64 でエンコードし SignatureValue 要素に格納する。

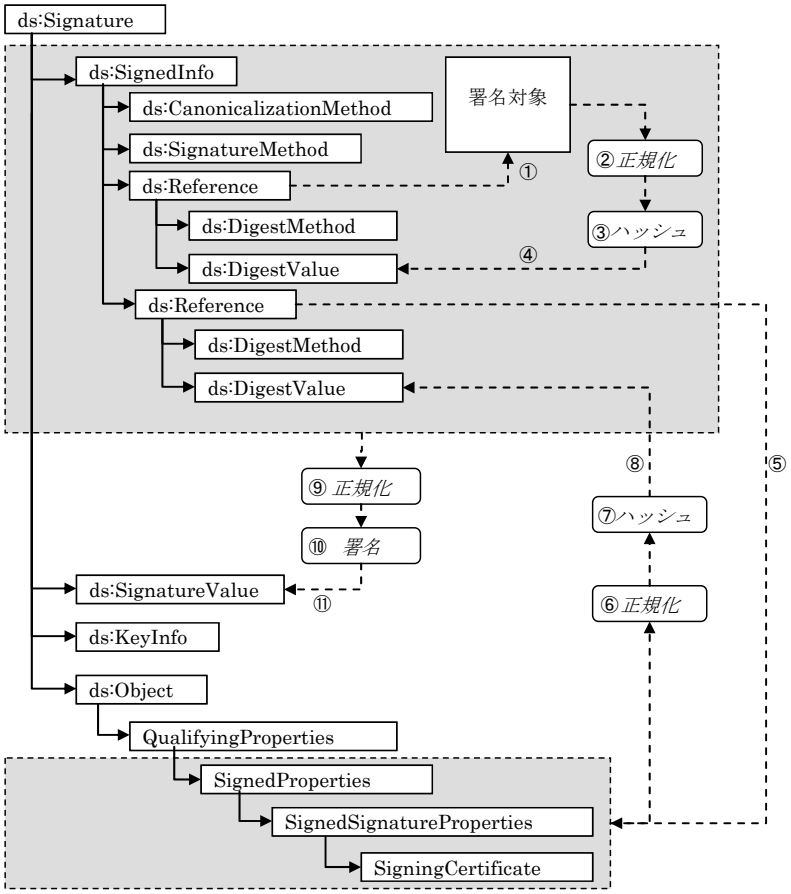


図 3.1.1.2.1：XAdES-BES の生成手順

3.1.2. XAdES-T の生成

XAdES-T は、署名値にタイムスタンプを付与することで署名時刻を確定するもので、XAdES-BES または XAdES-EPES をベースに生成する。図 3.1.2.1 に XAdES-T の生成手順の概要を、図 3.1.2.2 に SignatureTimeStamp の生成手順を示し、以下に図示する手順の詳細を示す。以下の③～⑦は図 3.1.2.2 の③～⑦に対応する。

①. QualifyingProperties 要素、UnsignedProperties 要素および UnsignedSignatureProperties 要素が存在しなれば追加する（CounterSignature 要素がある場合は、UnsignedProperties 要素および UnsignedSignatureProperties 要素は既に存在する）。

②. SignatureTimeStamp 要素を生成し UnsignedSignaturePropeties 要素に配置する。
③. ds:SignatureValue 要素をその内容（署名値）も含めて取り出す。
④. 取り出した要素を正規化する。正規化のアルゴリズムは、通常 XML 署名で標準的な正規化手法を用いる（一般的には、Canonical XML [33]）。別の正規化手法を利用いる場合は、SignatureTimeStamp 要素の配下の ds:CanonicalizationMethod 要素で用いた正規化手法を指定する。
⑤. 正規化された octet stream に対してハッシュ値を計算する。
⑥. ハッシュ値をタイムスタンプ局に送信し、ハッシュ値に対するタイムスタンプを取得する。なお、RFC 3161[5]に準拠したタイムスタンプを用いる場合は、ハッシュ値の計算に用いたハッシュ関数は、タイムスタンプリクエストに含めてタイムスタンプを取得する。
⑦. タイムスタンプ局から取得したタイムスタンプトークンは base64 でエンコードし、EncapsulatedTimeStamp 要素に格納する。EncapsulatedTimeStamp 要素は SingatureTimeStamp 要素の配下に配置する。

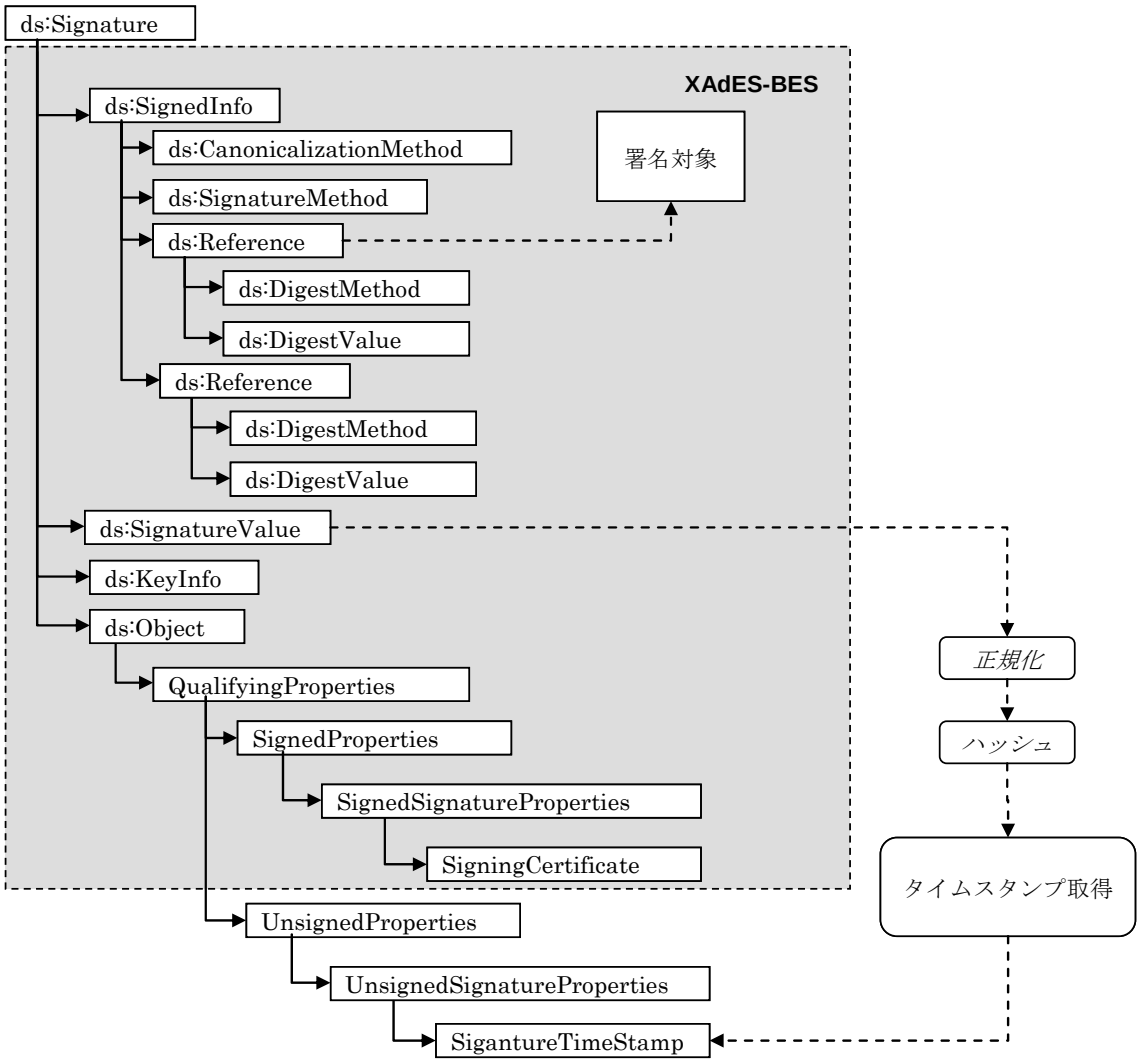


図 3.1.2.1：XAdES-T の生成手順

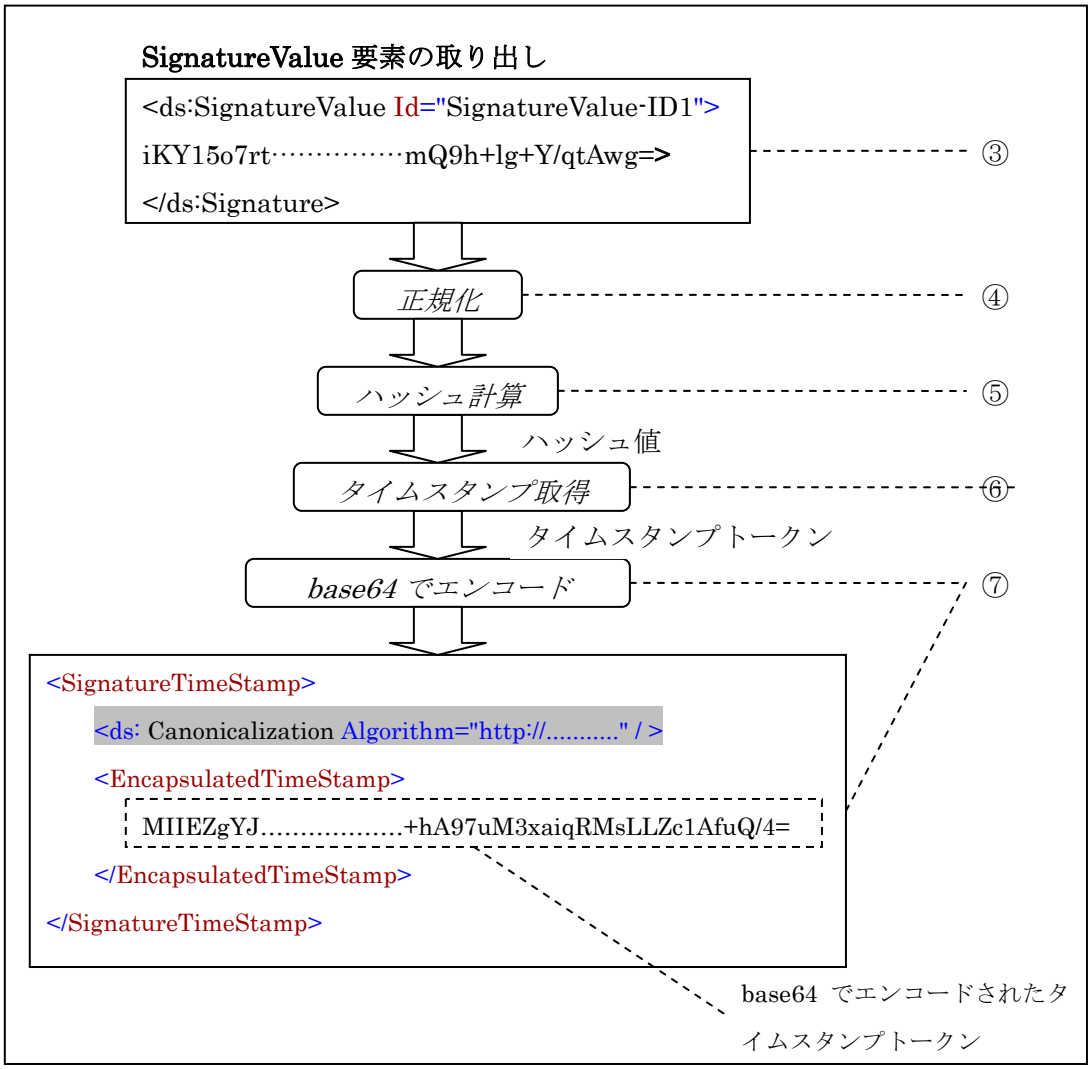


図 3.1.2.2：SignatureTimeStamp の生成手順

3.1.3. XAdES-C、XAdES-X、XAdES-X-L の生成

3.1.3.1. XAdES-C

XAdES-C は、XAdES-T をベースに生成する。図 3.1.3.1.1 に XAdES－C の生成手順の概要を示し、以下に図示する手順の詳細を示す。

①. 現在の時刻で署名者証明書を検証し、検証に必要な証拠情報（証明書チェーンや関連する失効情報）を収集する。猶予期間を考慮する場合は、署名タイムスタンプの時刻から猶予期間が経過した後に検証に必要な証拠情報を収集する。

②. 取得した証明書のうち署名者証明書を除いた証明書を元に UnsignedSignaturePropeties 配下に CompleteCertificateRefs 要素を生成する。

・ 証明書の発行者を IssuerSerial 要素の配下の ds:X509IssuerName 要素に格納

する。

- 証明書のシリアル番号を IssuerSerial 要素の配下の ds:X509 SerialNumber に格納する。
- 証明書のハッシュ値を計算し、計算結果を base64 でエンコードする。
- CertDigest 要素の配下の ds:DigestValue 要素にエンコードされたハッシュ値を、ds:DigestMethod 要素の Algorithm 属性にハッシュ関数名を格納する。

③. 取得した失効情報を元に UnsignedSignaturePropeties 配下に CompleteRevocationRefs 要素を生成する。

- CRL の発行者を Issuer 要素に、CRL の有効開始日（thisUpdate 属性）の日付を IssueTime 要素に、CRL 番号（CRLNumber）を Number 要素に格納する。
- CRL のハッシュ値を計算し、計算結果を base64 でエンコードする。
- CertDigest 要素の配下の ds:DigestValue 要素にエンコードされたハッシュ値を、ds:DigestMethod 要素の Algorithm 属性にハッシュ関数名を格納する。

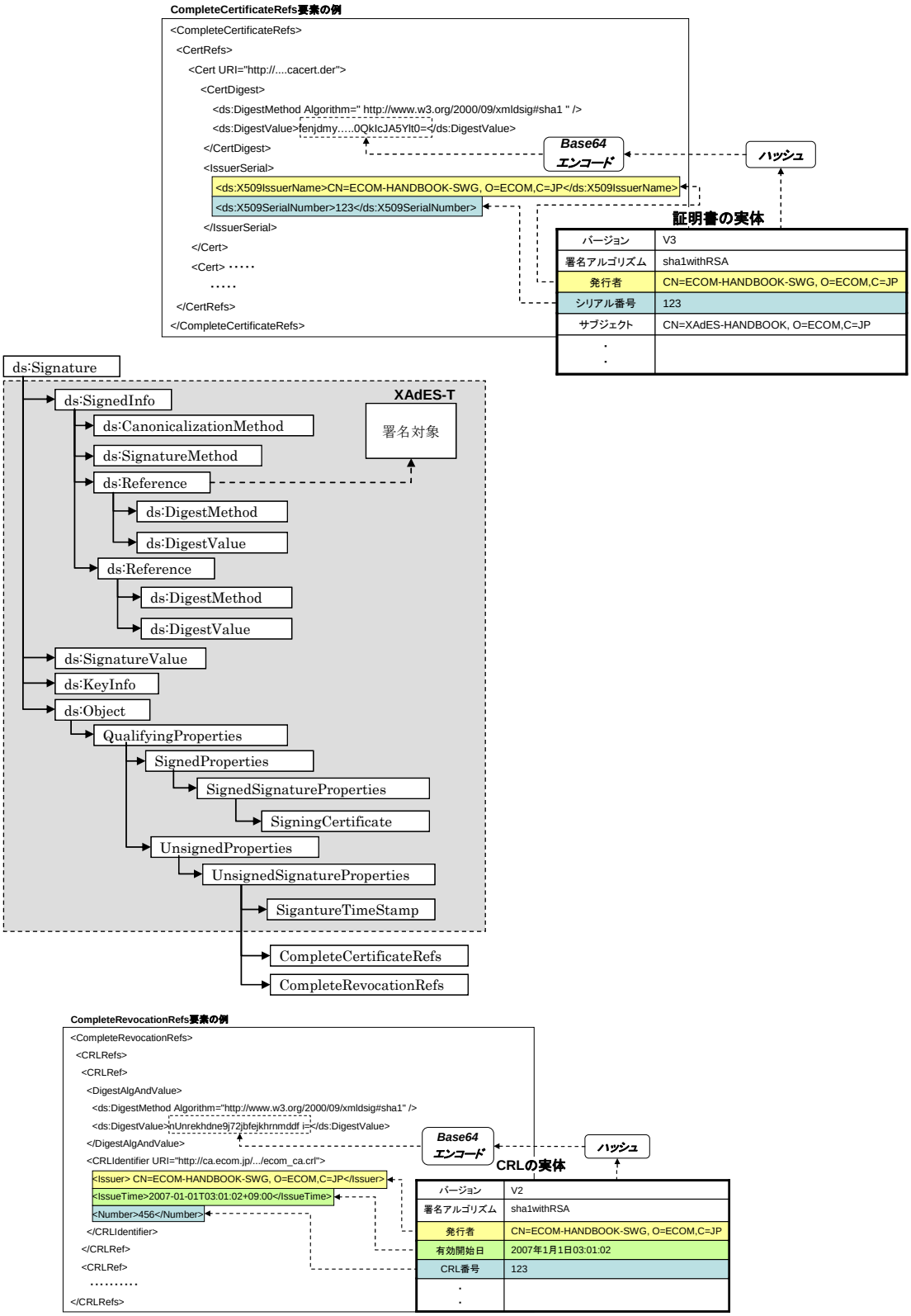


図 3.1.3.1.1：XAdES-C の生成手順

3.1.3.2. XAdES-X の生成

XAdES-X は、XAdES-C をベースに生成される。署名値や署名タイムスタンプと証明書や失効情報への参照情報に対しタイムスタンプ（SigAndRefsTimeStamp）を付与する XAdES-X type1 と、証明書や失効情報への参照情報のみに対しタイムスタンプ（RefsOnlyTimeStamp）を付与する XAdES-X type2 の二通りが定義されている。ここでは、not distributed case について説明する。図 3.1.3.2.1 に XAdES-X の生成手順の概要を示し、以下に図示する手順の詳細を示す。

①. RefsOnlyTimeStamp 要素（または SigAndRefsTimeStamp 要素）を生成し UnsignedSignaturePropeties 要素に配置する。

②. CompleteCertificateRefs 要素と CompleteRevocationRefs 要素（SigAndRefsOnlyTimeStamp を生成する場合は、ds:SignatureValue 要素、SignatureTimeStamp 要素も含む）をその内容（署名値）も含めて取り出す。

③. 取り出した要素を正規化する。正規化のアルゴリズムは、通常 XML 署名で標準的な正規化手法を用いる（一般的には、Canonical XML [33]）。別の正規化手法を利用する場合は、RefsOnlyTimeStamp 要素(または SigAndRefsTimeStamp 要素)の配下の ds:CanonicalizationMethod 要素で用いた正規化手法を指定する。

④. 正規化された octet stream を XAdES 署名フォーマット内に出現する順序に従い連結し、そのハッシュ値を計算する。

⑤. ハッシュ値をタイムスタンプ局に送信し、ハッシュ値に対するタイムスタンプを取得する。なお、RFC 3161[5]に準拠したタイムスタンプを用いる場合は、ハッシュ値の計算に用いたハッシュ関数は、タイムスタンプリクエストに含めてタイムスタンプを取得する。

⑥. タイムスタンプ局から取得したタイムスタンプトークンは base64 でエンコードし、EncapsulatedTimeStamp 要素に格納する。EncapsulatedTimeStamp 要素は RefsOnlyTimeStamp 要素(または SigAndRefsTimeStamp 要素)の配下に配置する。

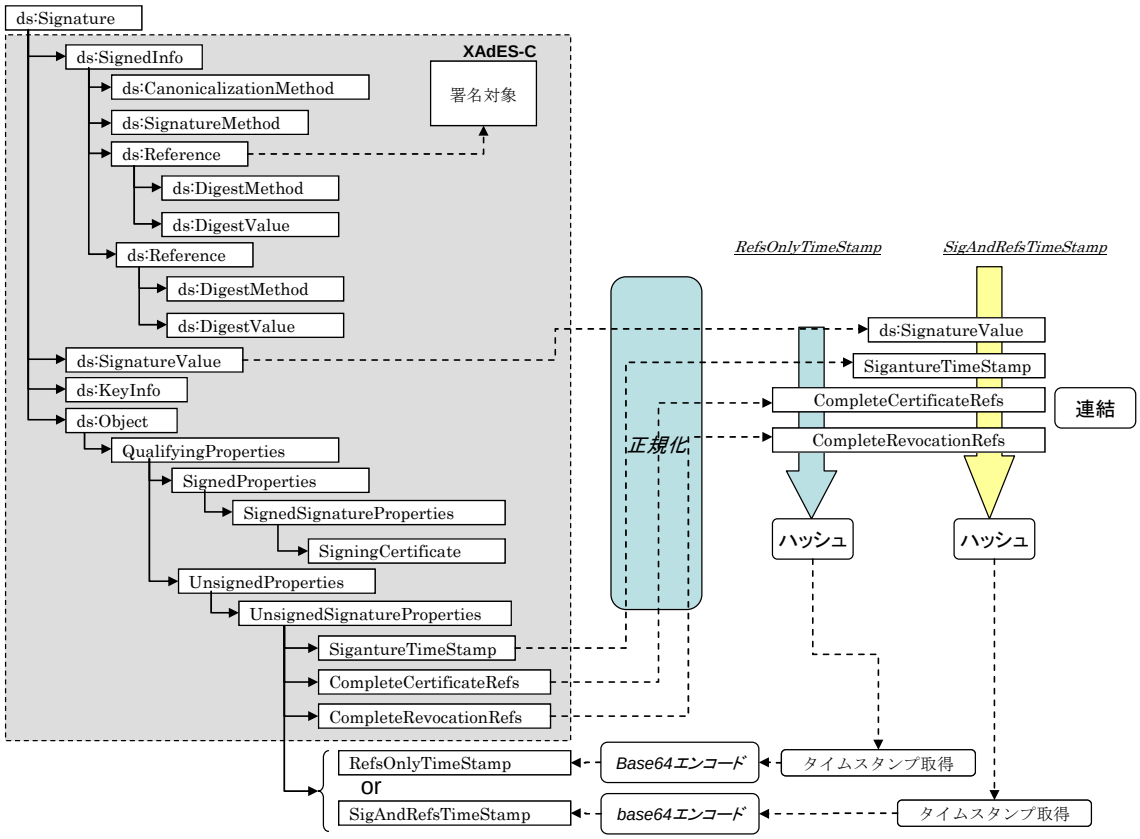


図 3.1.3.2.1：XAdES-X の生成手順

3.1.3.3. XAdES-X-L の生成

XAdES-X-L は、XAdES-X type1 または XAdES-X type2 をベースに生成する。証明書や失効情報の参照情報で参照されている証明書や失効情報を追加する。図 3.1.3.3.1 に XAdES-X-L の生成手順の概要を示し、以下に図示する手順の詳細を示す。

①. CertificateValues 要素および RevocationValues 要素を生成し UnsignedSignaturePropeties 要素に配置する。

②. CompleteCertificateRefs 要素、CompleteRevocationRefs 要素で参照されている証明書及び CRL や OCSP などの失効情報を収集する。

③. 収集した失効情報を base64 でエンコードし、CompleteCertificateRefs 要素、CompleteRevocationRefs 要素に格納する。

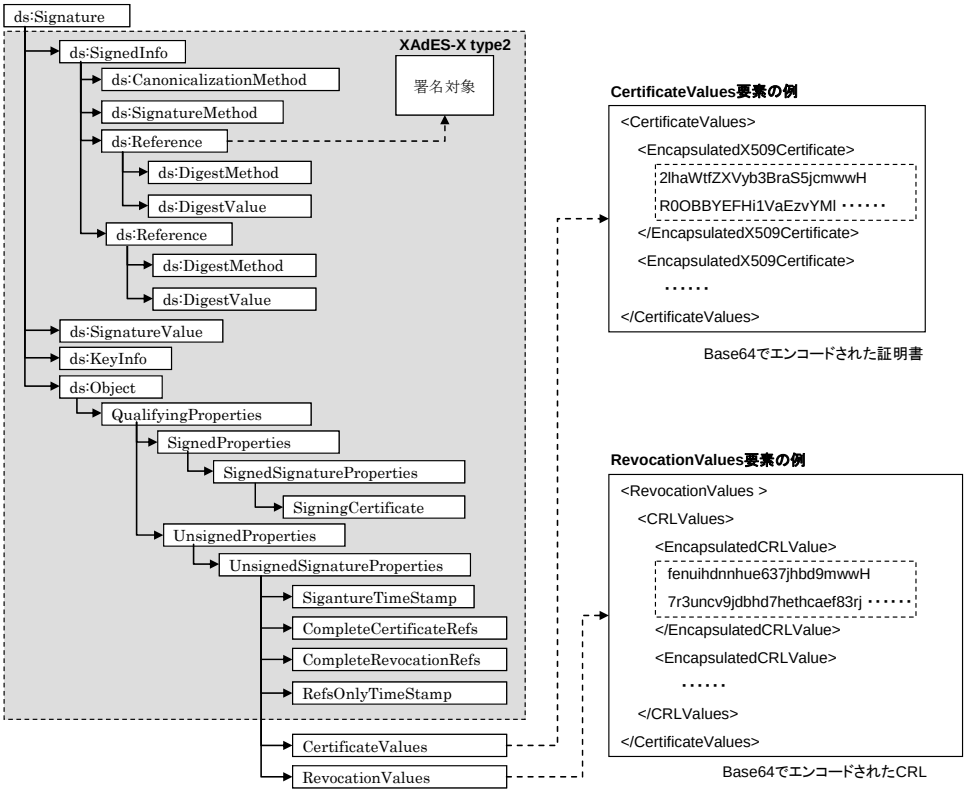


図 3.1.3.3.1：XAdES-X-L の生成手順

3.1.4. 第一世代 XAdES-A の生成

XAdES-A は、デジタル署名の有効性を長期間保存できる形式で、XAdES-T または XAdES-X-L(-Type1 or Type2)などベースに生成する。ここでは、XAdES-T をベースに not distributed case の XAdES-A を生成する方法を説明する。図 3.1.4.1 に XAdES-A の生成手順の概要を、図 3.1.4.2 に ArchiveTimeStamp の生成手順を示し、以下に手順の詳細を示す。なお、以下の⑥～⑭は図 3.1.4.2 の⑥～⑭に対応する。

①. 最初に XAdES-T を検証する。SignatureTimeStamp の時刻で署名検証および署名者証明書検証および SignatureTimeStamp 要素に格納されたタイムスタンプトークンを検証する。

②. 現在の時刻で署名者証明書を検証し、検証に必要な証拠情報（証明書チェーンや関連する失効情報）を収集する。猶予期間を考慮する場合は、署名タイムスタンプの時刻から猶予期間が経過した後に検証に必要な証拠情報を収集する。

③. UnsignedSignaturePropeties 配下に CertificateValues 要素を生成し、収集した証明書チェーンをその配下に格納する。なお、Certificagevalues 要素には、署名者証明書に検証に必要な証明書チェーンを構成するすべての証明書を格納する（署名者証明書を含む）。ただし、ds:KeyInfo 要素に含まれている証明書はここに含めなくても良い。CompleteCertificateRefs 要素が存在する場合、そこで参照されるすべ

ての失効情報の実体を CertificateValues に含まなければならない（MUST [22] 7.6.1)。

④. 同様に UnsignedSignaturePropeties 配下に RevocationValues 要素を生成し、収集したすべての失効情報を格納する。CompleteRevocationRefs 要素が存在する場合、そこで参照されるすべての失効情報の実体を RevocationValues に含まなければならない（MUST [22] 7.6.2)。

⑤. 一つ目のアーカイブタイムスタンプを付与する場合は署名タイムスタンプのタイムスタンプトークンを検証し、タイムスタンプの検証に必要な証拠情報（証明書チェーンと CRL 等の失効情報）を収集する。収集した証拠情報は、別途安全に保管するか SignatureTimeStamp 要素に格納されているタイムスタンプトークンの中に格納する（base64 をデコードし、タイムスタンプトークンに証拠情報を格納後、再エンコードする)。

⑥. ds:SignedInfo 要素内の ds:Reference 要素で参照しているコンテンツを取り出す。ここで取り出されるのは、署名対象データ、ds:KeyInfo 要素、SignedSignatureProperties 要素などである

⑦. 取り出した要素を正規化し、正規化の結果を連結する。正規化のアルゴリズムは、通常 XML 署名で標準的な正規化手法を用いる(一般的には、Canonical XML [33])。別の正規化手法を利用する場合は、ArchiveTimeStamp 要素の配下の ds:CanonicalizationMethod 要素で用いた正規化手法を指定する。

⑧. 以下の XML 署名に関する要素のそれぞれに対して以下の順番で取り出す。取り出したノードセットを正規化し、正規化の結果を連結する。

- ds:SignedInfo
- ds:SignatureValue
- ds:KeyInfo（存在した場合のみ）

⑨. UnsignedSignatureProperties 配下の、SignatureTimeStamp 要素、CertificateValues 要素、RevocationValues 要素および既に存在している ArchiveTimeStamp 要素をそれぞれ取り出す（UnsignedSignatureProperties 配下に、CounterSignature 要素や証拠情報への参照情報、証拠情報への参照情報に関するタイムスタンプ、属性証明書に関する情報なども存在する場合はそれらも処理対象となる)。

⑩. 取り出した要素を正規化する。正規化のアルゴリズムは、通常 XML 署名で標準的な正規化手法を用いる（一般的には、Canonical XML [33])。別の正規化手法を利用する場合は、ArchiveTimeStamp 要素の配下の ds:CanonicalizationMethod 要素で用いた正規化手法を指定する。

⑪. ⑦⑧⑩の結果、正規化されたデータを連結する。

⑫. 連結されたデータ（octet stream）に対してハッシュ値を計算する。

⑬. ハッシュ値をタイムスタンプ局に送信し、ハッシュ値に対するタイムスタンプを取得する。なお、RFC 3161[5]に準拠したタイムスタンプを用いる場合は、ハッシュ値の計算に用いたハッシュ関数は、タイムスタンプリクエストに含めてタイムスタンプを取得する。

⑭. タイムスタンプ局から取得したタイムスタンプトークンはbase64でエンコードし、EncapsulatedTimeStamp 要素に格納する。EncapsulatedTimeStamp 要素はArchiveTimeStamp要素の配下に配置する。

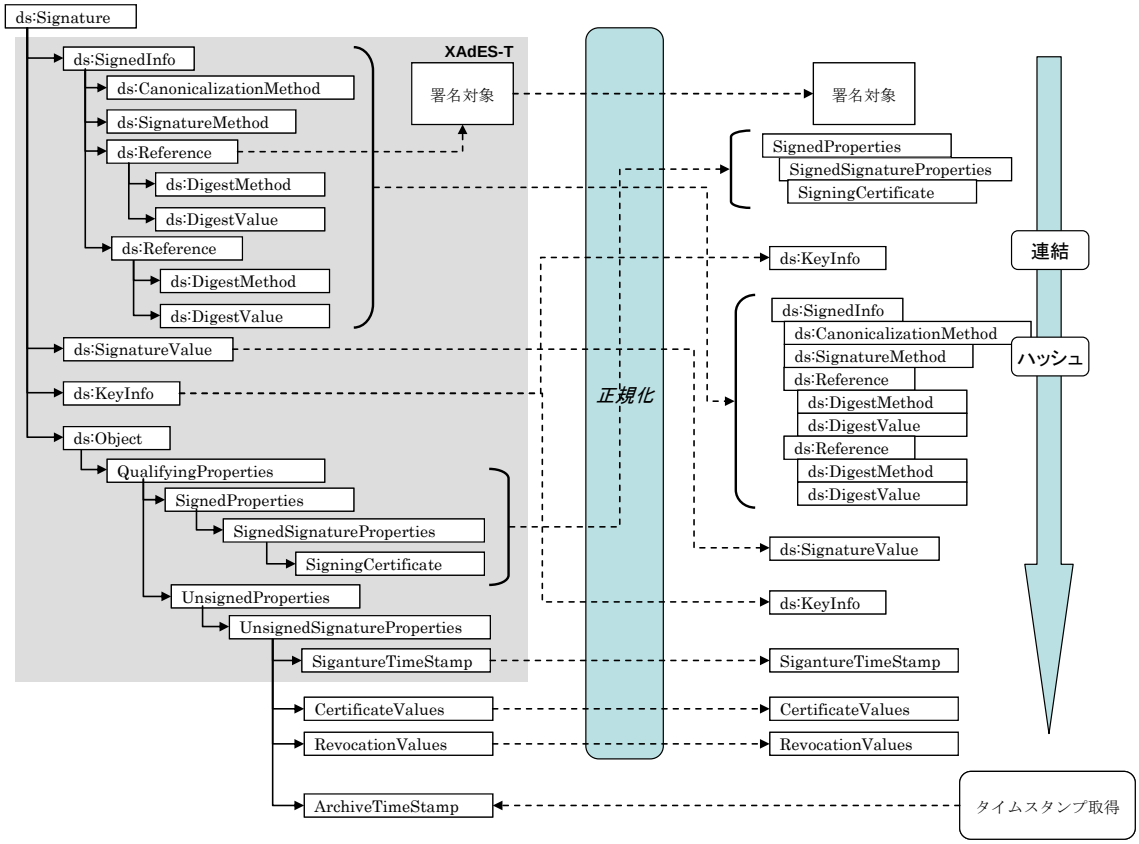


図 3.1.4.1：XAdES-A の生成手順

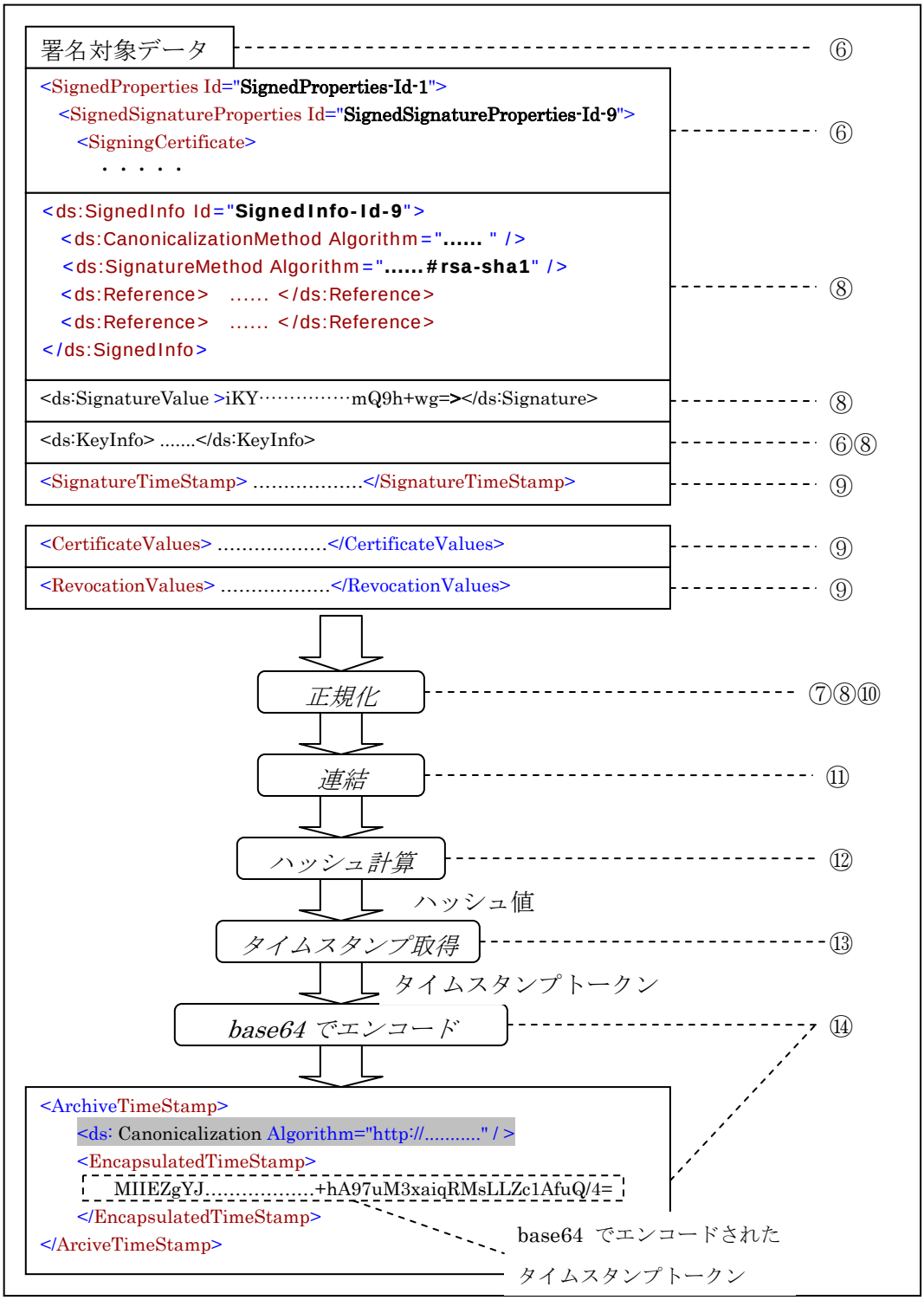


図 3.1.4.2：ArchiveTimeStamp 要素の生成方法

3.1.5. 第二世代 XAdES-A の生成

過去の XAdES-A フォーマットを、暗号アルゴリズムの危殆化および直前のアーカイブタイムスタンプの TSA 証明書の期限切れから保護するために、追加のアーカイブスタン

プを付与することにより第二世代以降の XAdES-A フォーマットが生成される。なお、アーカイブタイムスタンプを追加する前に、タイムスタンプの検証に必要な証拠情報（証明書チェーンと CRL 等の失効情報）を収集する。収集した証拠情報は、別途安全に保管するか直前の ArchiveTimeStamp 要素に格納されているタイムスタンプトークンの中に格納する必要がある（base64 をデコードし、タイムスタンプトークンに証拠情報を格納後、再エンコードする）。

3.2. XAdES フォーマットデータの生成における注意事項

3.2.1. タイムスタンプトークンの証拠情報の扱い

XAdES の標準仕様では、SignatureTimeStamp や ArchiveTimeStamp の検証情報（証明書チェーンや CRL、OCSP）の格納方法について言及されていない。しかし、RFC 3161[5]に準拠したタイムスタンプを利用する場合、何らかの方法でタイムスタンプトークン自体の検証情報を保存する必要がある。保存方法としては、以下のような方法がある。

- SignatureTimeStamp の場合
  1) タイムスタンプトークン内の unsigned attribute
  2) タイムスタンプトークン内の certificates および crls 属性
  3) XAdES を構成する CertificateValues や RevocationValues
  4) 別途、安全な方法で保管する。
- ArchiveTimeStamp の場合
  1) タイムスタンプトークン内の unsigned attribute
  2) タイムスタンプトークン内の certificates および crls 属性
  3) 別途、安全な方法で保管する。

第一部 デジタル署名ハンドブック

III. デジタル署名の検証

1. CAdES、XAdES に共通の事項

本章では長期署名フォーマットの一般的な検証方法を述べる。以降、CAdES-*とXAdES-*を総称して ES-*と記述する。ES、ES-T、ES-C、ES-X (type1,type2)、ES-X Long、ES-A の各タイプについて、それぞれ単独のフォーマットとして運用する場合を想定して各々に必要な検証の要件を述べる。ここでは主に長期署名フォーマットで必須として定義されている各項目に対する検証方法について記述し、オプションとして定義されている拡張要素の詳細は扱わない。オプション要素はデータ生成側と検証側が合意のもとで別途規定した定義やルールが必要である。

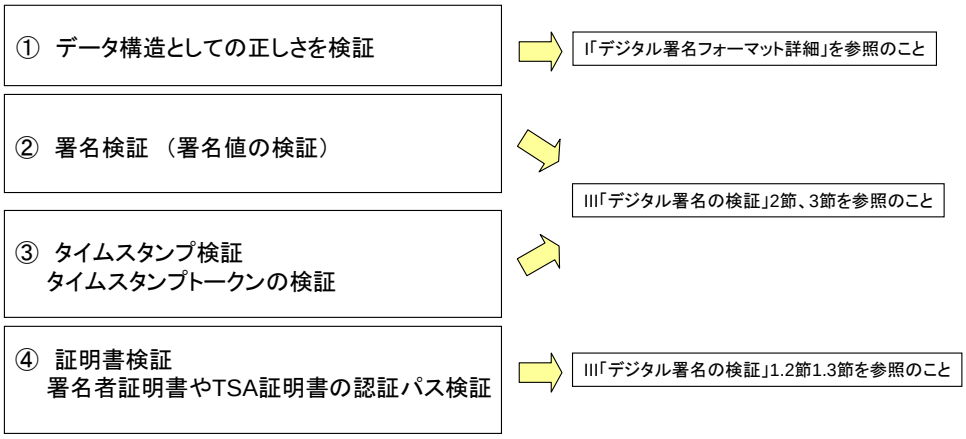
1.1 節では基本的な検証項目を大別し、フォーマットのタイプ別に必要な検証項目を記述する。これらの検証項目に対して CAdES における具体的な詳細を 2 節に、XAdES における詳細を 3 節に示している。

1.2 節では証明書検証に必要な証明書や失効情報の取得方法を示し、1.3 節では証明書検証で指定する時刻パラメータの扱いについて考察している。

なお、タイムスタンプは RFC 3161 タイムスタンプを対象としている。RFC 3161 とは異なるタイムスタンプを使用する場合には、別途規定された方法に従い検証を行うこと。

1.1. 長期署名フォーマットの検証項目

長期署名フォーマットの検証で確認する事項を大別すると以下の工程になる。



① データ構造の検証

長期署名フォーマットのデータとして正しい構造であることを確認する。

② 署名検証

署名対象との一致確認や公開鍵を用いた署名検証を行う。

③ タイムスタンプ検証

タイムスタンプトークンの検証を行う。

④ 証明書の認証パス検証

②や③の検証で用いた署名者証明書やタイムスタンプの TSA 証明書の認証パス検証を行う。

この章では①データ構造の検証の詳細については記述せず、検証対象の署名データは正しいフォーマットであることを前提とする。また、④証明書の認証パス検証の詳細についても対象外とし、長期署名フォーマットの検証に必要な証明書や失効情報の取得方法(1.2 節)や、証明書の有効性を確認する時刻情報(1.3 節)についてのみ記述する。

表 1.1.1 にフォーマットのタイプ別に検証を行う項目をまとめる。

各検証項目に対応した詳細を 2 節(CAdES)と 3 節(XAdES)に記述している。

表 1.1.1 各フォーマットタイプの検証要件

	ES	ES-T	ES-C	ES-X type1,type2	ES-X Long	ES-A
署名者の署名検証	必須	必須	必須	必須	必須	必須
署名タイムスタンプの検証	-	必須	必須	必須	必須	必須
証明書・失効情報の参照情報に対する一致確認	-	-	必須	必須	必須	オプション
type1,type2 タイムスタンプの検証	-	-	-	必須	-	オプション
アーカイブタイムスタンプの検証	-	-	-	-	-	必須

CAdES-X type1,type2 とは検証情報を保護するタイムスタンプが付与された形式であり、XAdES では RefOnlyTimeStamp, SigAndRefsTimeStamp のタイムスタンプを指している。

1.2. 検証に必要な証明書と失効情報の取得

この節では長期署名フォーマットのタイプ毎に証明書検証に必要な証明書や失効情報の取得方法について記述する。

証明書の認証パス構築・認証パス検証では以下の検証情報が必要である。

・トラストアンカとなる証明書
・認証パスを構成する中間 CA 証明書
・エンドエンティティの証明書（署名者証明書、TSA 証明書）
・CA が発行する失効情報
　トラストアンカの CA 及び認証パス上の中間 CA が発行する失効情報
・失効情報の署名に用いられた証明書

現状の長期署名フォーマットでは、その内部に格納された証明書群からトラストアンカとなる証明書を特定する手段は無く、検証時にはトラストアンカとなる証明書を指定し、その証明書までの認証パスを検証する必要がある。トラストアンカとなる証明書は署名者と検証者が合意のもとで事前に決定されるものである。署名ポリシにもとづく署名を検証する場合には署名ポリシによって明示されたトラストアンカを指定する。

認証パスを構成する証明書と失効情報の取得方法は以下のいずれかとなる。

- 検証情報が署名データ内にアーカイブされている場合
  検証情報を保護するタイムスタンプ(type1, type2)やアーカイブタイムスタンプによって検証情報が保護されている場合には、それらの検証情報を用いて検証する。
- 検証情報が署名データ内にアーカイブされていない場合
  認証局や信頼できる第三者機関より取得する。検証時点において認証局が発行する最新の検証情報が取得可能である場合や、信頼できる第三者機関によって署名データ生成当時における有効性を確認できる検証情報が取得可能である場合などが考えられる。

1.3. 証明書の検証時刻の指定について

長期署名フォーマットには署名者の証明書、署名タイムスタンプを行う TSA の証明書、アーカイブタイムスタンプを行う TSA の証明書など様々な証明書が含まれる。これらの証明書を検証するには、それぞれのトラストアンカとなるルート証明書までの証明書チェーンを辿り、有効期限、失効状態、証明書や CRL などの拡張領域を確認する必要がある。長期署名フォーマットにおいては、検証時刻、即ち各証明書が何時の時点で有効であったのか確認することが必要である。フォーマット形式(ES,ES-T,ES-C,ES-X Long,ES-A)ごとに、証明書の有効期限や失効状態を確認するための基点となるそれぞれの時刻について解説する。

1.3.1. ES における証明書の有効性を確認する時刻

ES 単体での利用では署名時刻が確定されていないため、署名者証明書は検証する時刻における有効性を確認する。したがって、たとえ署名した時点では署名者証明書が有効であったとしても、検証する時点で署名者の証明書の有効性が無くなっていた場合には、署名自体の有効性も無くなってしまう。署名時刻が保証されない ES のそもそもの問題点である。

1.3.2.　ES-T における証明書の有効性を確認する時刻

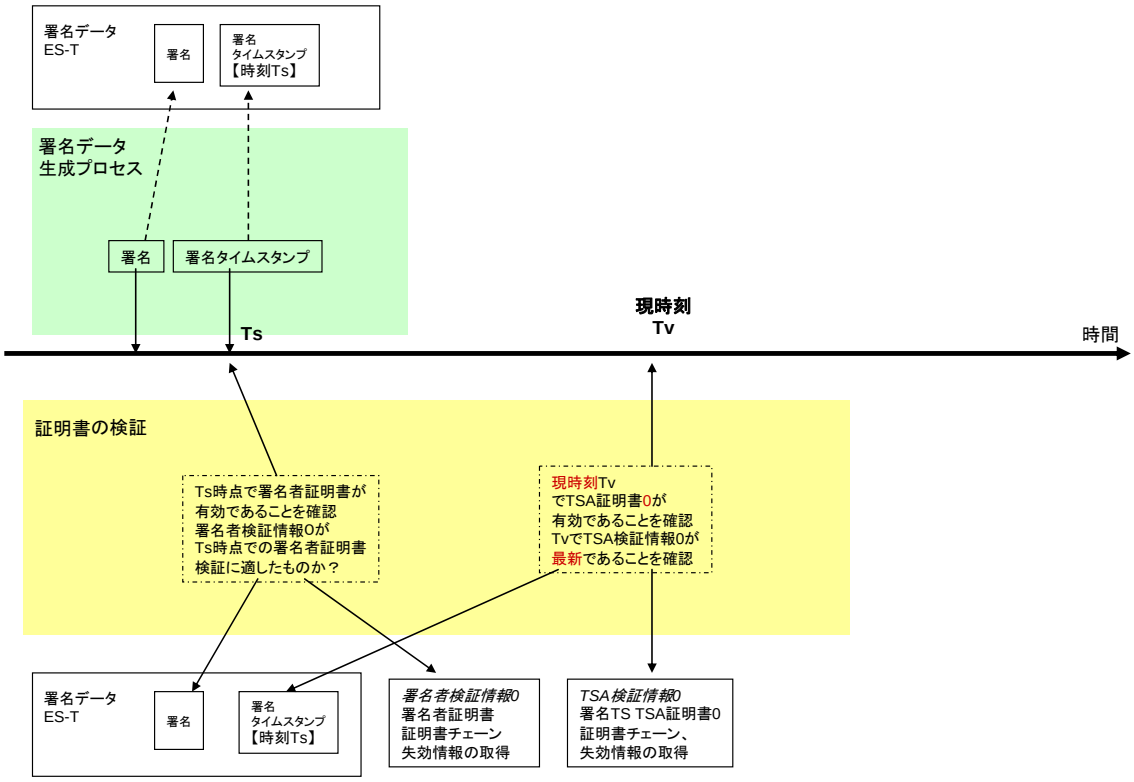


図 1.3.2.1 ES-T における証明書の有効性を確認する時刻

図 1.3.2.1 は署名データ生成時のプロセスと、検証に用いる時刻情報の対応関係を示したものである。

署名者証明書の検証では署名者が署名を付した当時の有効性を確認する。

すなわち、

- 署名タイムスタンプの時刻(Ts)が証明書の有効期限内にあること
- Ts の時点で証明書が失効されていないこと
- 失効検証を行うための検証情報が時刻 Ts での証明書検証に適したものであること
  （失効情報が発行された時間が適切であるか、失効情報の署名に用いられた CA 証明書の有効期限は適切であるか等）

を確認する。

署名タイムスタンプの TSA 証明書については検証時点での有効性を確認する。

すなわち、

- 検証時に TSA 証明書が有効期限内にあること
- 検証時に TSA 証明書が失効されていないこと
- 失効検証を行うための検証情報が検証時点で最新のものであること

を確認する。

1.3.3.　ES-C,ES-X Long における証明書の有効性を確認する時刻

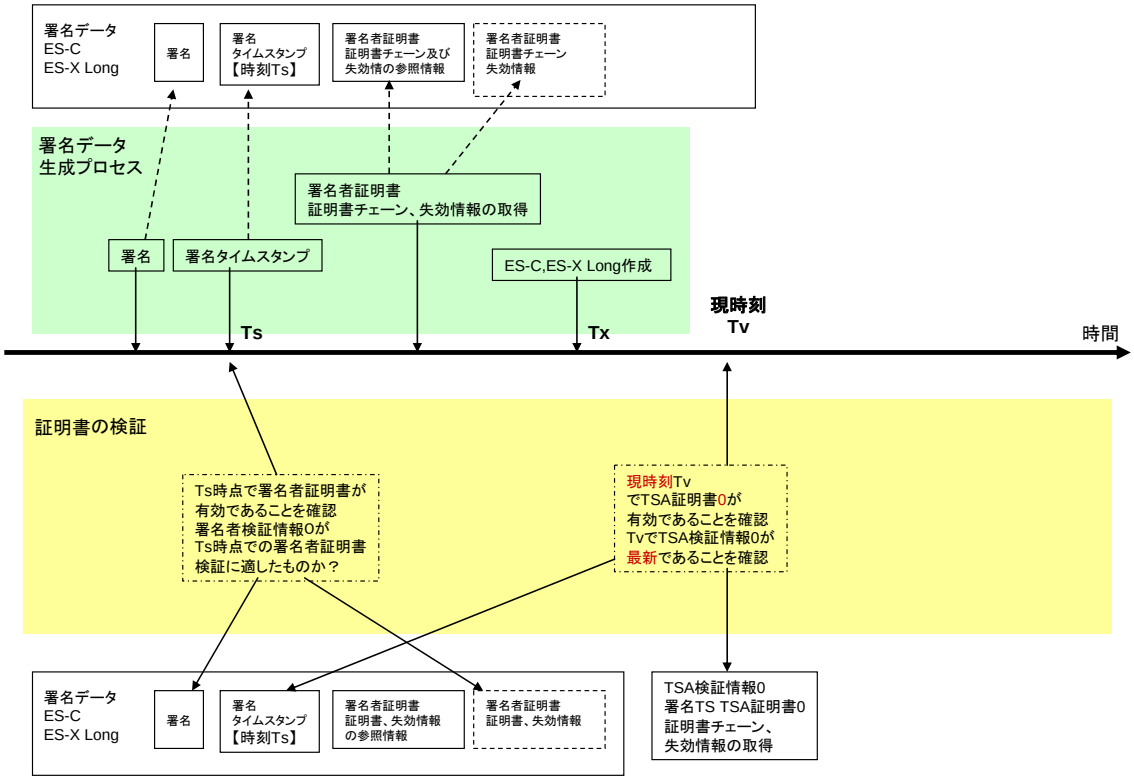


図 1.3.3.1 ES-C,ES-X Long における証明書の有効性を確認する時刻

type1 や type2 といった検証情報を保護するタイムスタンプを持たない ES-C 形式,ES-X Long 形式についての検証時刻をまとめたものが図 1.3.3.1 である。

1.3.4.　ES-X type1,type2 における証明書の有効性を確認する時刻

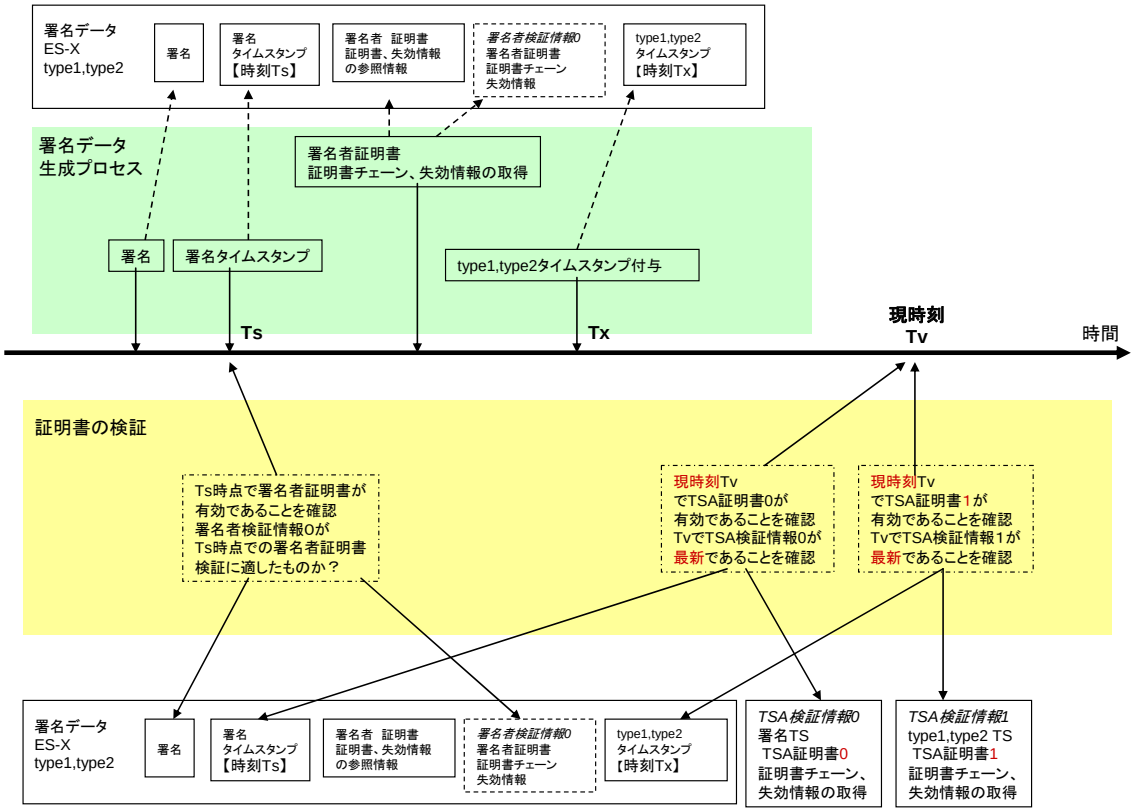


図 1.3.4.1 ES-X type1,type2 における証明書の有効性を確認する時刻

ES-X type1,type2 形式についての検証時刻をまとめたものが図 1.3.4.1 である。type1,type2 タイムスタンプの TSA 証明書は署名タイムスタンプの場合と同様に検証時点での有効性を確認する。

1.3.5. ES-A における証明書の有効性を確認する時刻

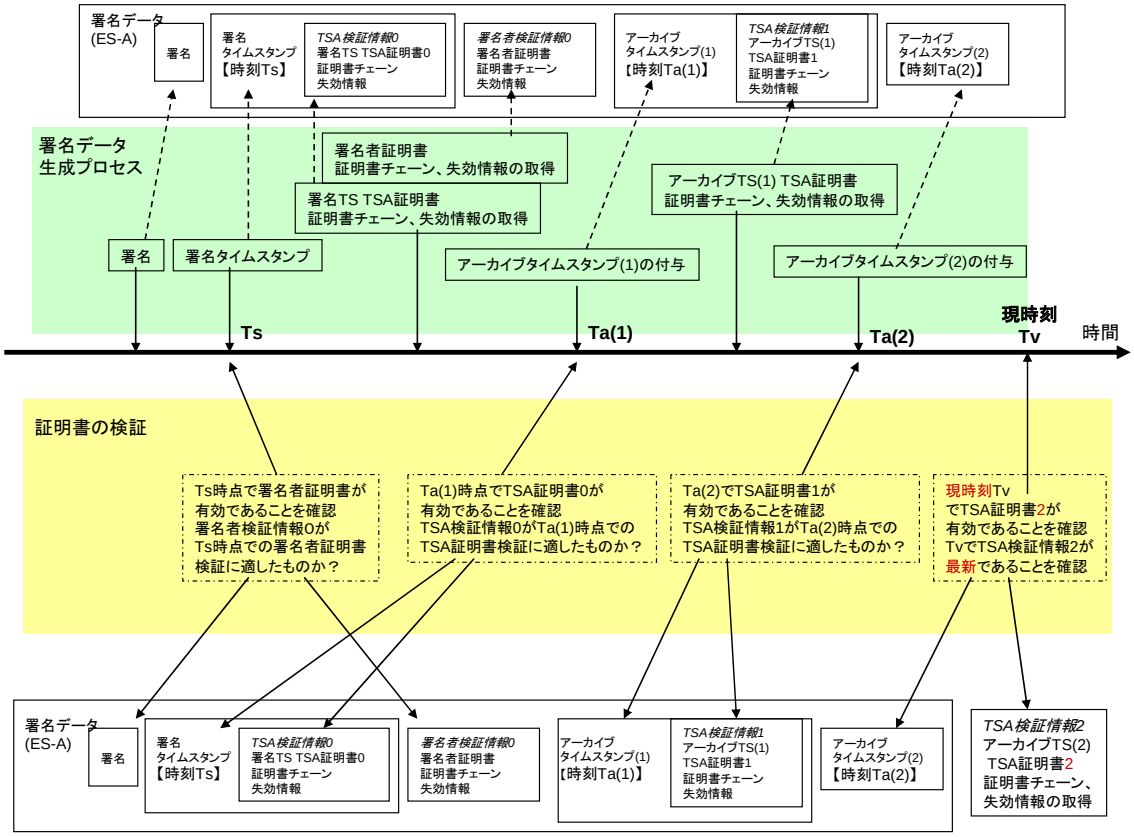


図 1.3.5.1 ES-A における証明書の有効性を確認する時刻(2 世代目の場合)

ES-A 形式についての検証時刻をまとめたものが図 1.3.5.1 である。図は例としてアーカイブタイムスタンプが更新された 2 世代目のアーカイブタイムスタンプを検証する場合について記述している。

- 署名者証明書は署名時刻 Ts での有効性を確認する。
- 署名タイムスタンプの TSA 証明書は 1 世代目のアーカイブタイムスタンプの時刻(図中の Ta(1))での有効性を確認する。
- 最新でないアーカイブタイムスタンプ（Ta(i)）の TSA 証明書は、そのアーカイブタイムスタンプを更新したアーカイブタイムスタンプの時刻(Ta(i+1))での有効性を確認する。
- 最新のアーカイブタイムスタンプの TSA 証明書は検証時点での有効性を確認する。

1.3.6. 検証時刻まとめ

1.3.1 節から 1.3.5 節で述べた証明書の検証時刻を表 1.3.6.1 にまとめる。

表中の時刻を表す記号は以下の通りである。

Ts: 署名タイムスタンプの時刻

Tv: 検証処理を実行した時刻

Tx: ES-X type1,type2 の時刻

Ta(i): i 世代のアーカイブタイムスタンプの時刻（i=1 が最も古いタイムスタンプ)

-: 該当なし

表 1.3.6.1

	ES	ES-T	ES-C	ES-X Long	ES-A	第二世代以降の ES-A
署名者証明書の有効性確認	Tv	Ts	Ts	Ts	Ts	Ts
署名タイムスタンプの有効性確認	Tv	Tv	Tx/Tv	Tx/Tv	Ta(1)	Ta(1)
i 世代のアーカイブタイムスタンプ TSA 証明書の有効性確認（最新ではないアーカイブタイムスタンプ）	-	-	-	-	-	Ta(i+1)
最新のアーカイブタイムスタンプ TSA 証明書の有効性確認	-	-	-	-	-	Tv

1.3.7. 失効情報取得の猶予期間

証明書を失効させる場合には認証局に対して失効申請を行うが、実際に失効情報へ反映されるにはある程度の時間を要する。認証局が失効申請を受けてから実際に失効情報へ反映させるまでの期間（猶予期間,Grace Period）が署名検証において問題になるケースがある。

例えば署名者証明書の場合、正規の証明書所持者が鍵を紛失する等の理由により証明書の失効申請を出すとき、それとほぼ同時期に鍵を不正入手した悪意ある者が署名を偽造したとする。署名の検証者は猶予期間を経過しない状況で失効情報を取得した場合、偽造された署名に使われた証明書は失効されていないため正規の署名であると判断してしまう（図 1.3.7.1)。このようなケースでは猶予期間が経過した後の失効情報を確認すれば署名の不正を検知することが出来る（図 1.3.7.2)。

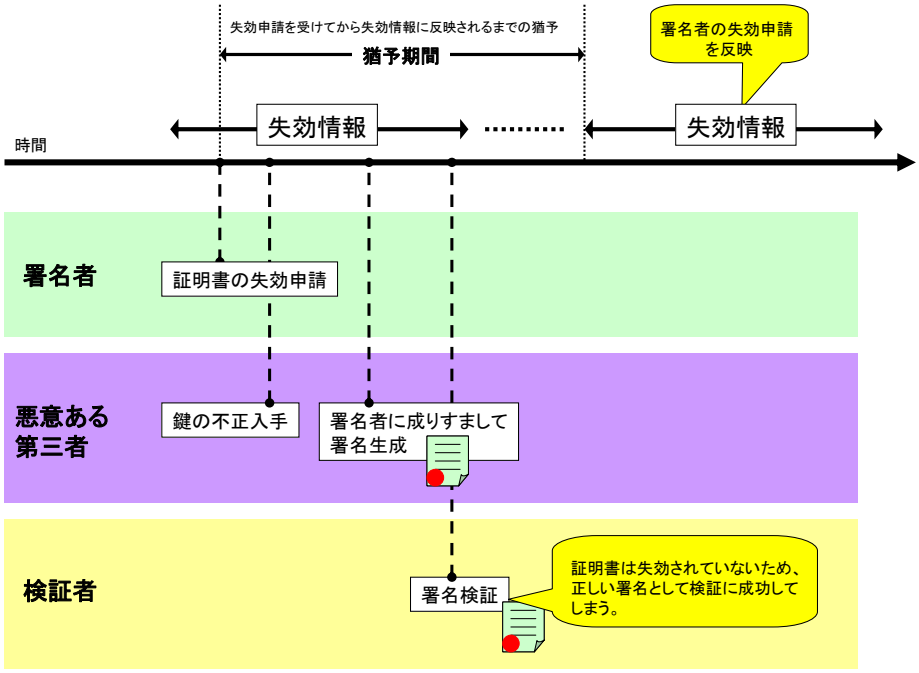


図 1.3.7.1 失効が反映されない失効情報で問題が生じるケース（猶予期間なし）

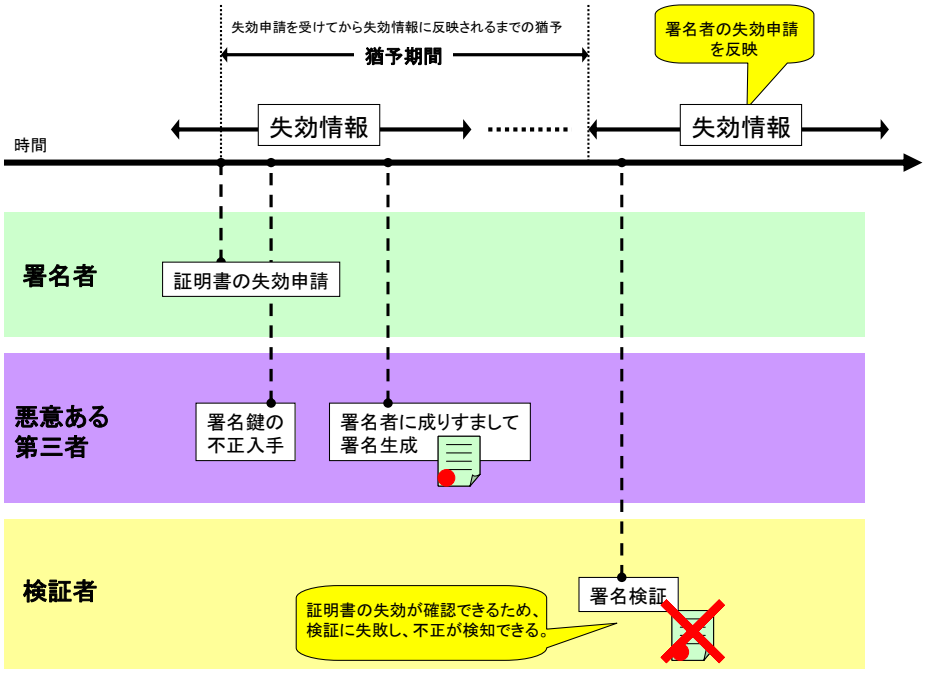


図 1.3.7.2 失効が反映された失効情報で検証するケース（猶予期間あり）

猶予期間を考慮した失効検証は検証対象となる証明書（署名者証明書、TSA 証明書）の運用方法やその証明書を発行する認証局の運用ポリシに照らして署名生成者と署名検証者の合意の元で、その妥当性を十分に判断したうえで実施することになる。

2. CAdES

この節では表 1.1.1 に示した各項目の詳細を述べる。各フォーマットの検証において該当する検証項目を参照のこと。

2.1. 署名者の署名検証

長期署名フォーマット中の signerInfo に対して下記の項目を検証する。

下記の項目のいずれかの検証に失敗した場合には、その署名は無効であるとみなす。

- MessageDigest 属性の一致確認
- 署名者証明書の一致確認
- 署名者証明書（公開鍵）による署名値の検証
- 署名者証明書のパス構築・パス検証

複数の signerInfo が存在する場合には、各々の signerInfo に対して検証する。

① MessageDigest 属性の一致確認

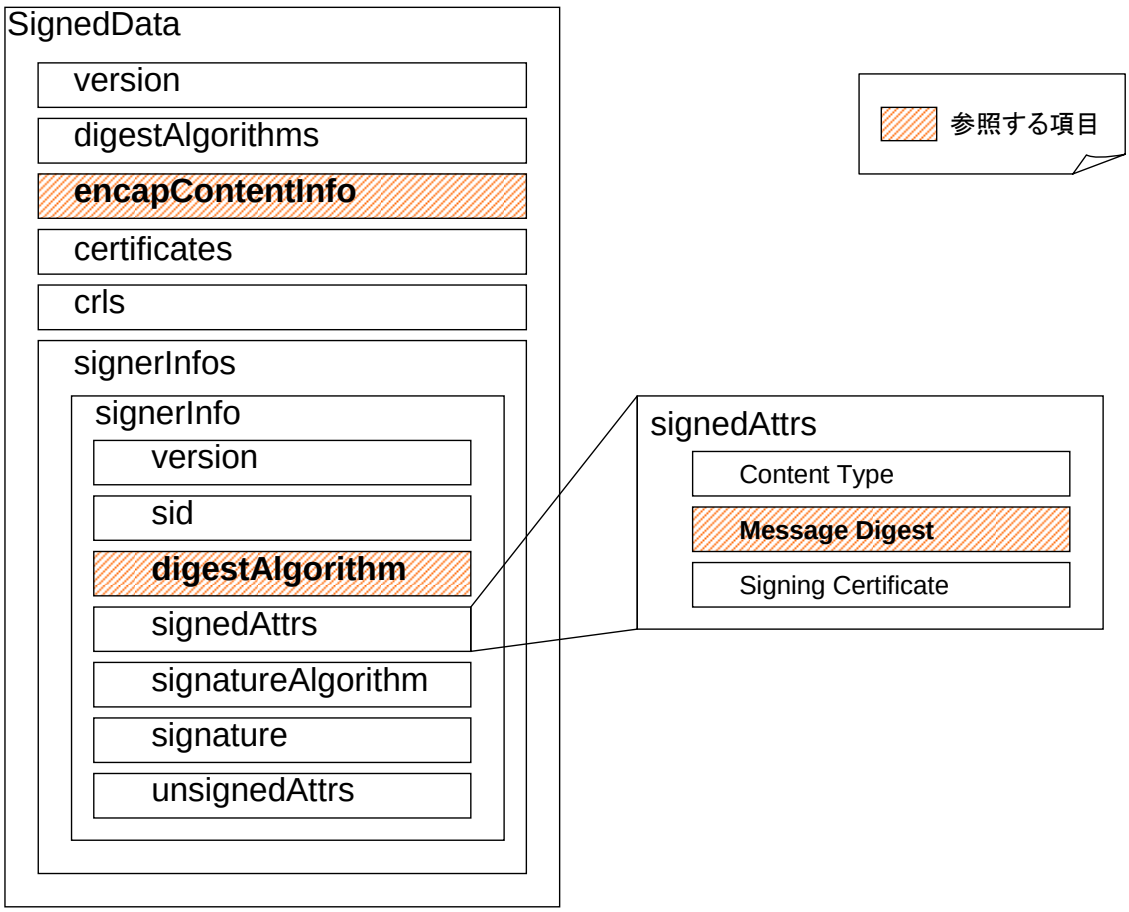


図 2.1.1 MessageDigest 属性の一致確認で参照する項目

signerInfo の署名属性 MessageDigest と、署名対象のデータとの一致を以下の手順で確認する。

手順	1. eContent データ(*1)を signerInfo digestAlgorithm フィールドで指定されたハッシュアルゴリズムによってハッシュ計算する。 2. MessageDigest の属性値である OCTET STRING からタグ(T)と長さ(L)を除いた値(V)のデータを取り出す。 3. 1.と 2.の値を比較し、一致していることを確認する。

(*1) 内包署名の場合は、encapContentInfo eContent フィールドの OCTET STRING からタグ(T)と長さ(L)を除いた値(V)のデータ。分離署名の場合は、長期署名フォーマットの外部に置かれた、eContent に相当するデータ。

② 署名者証明書の一致確認

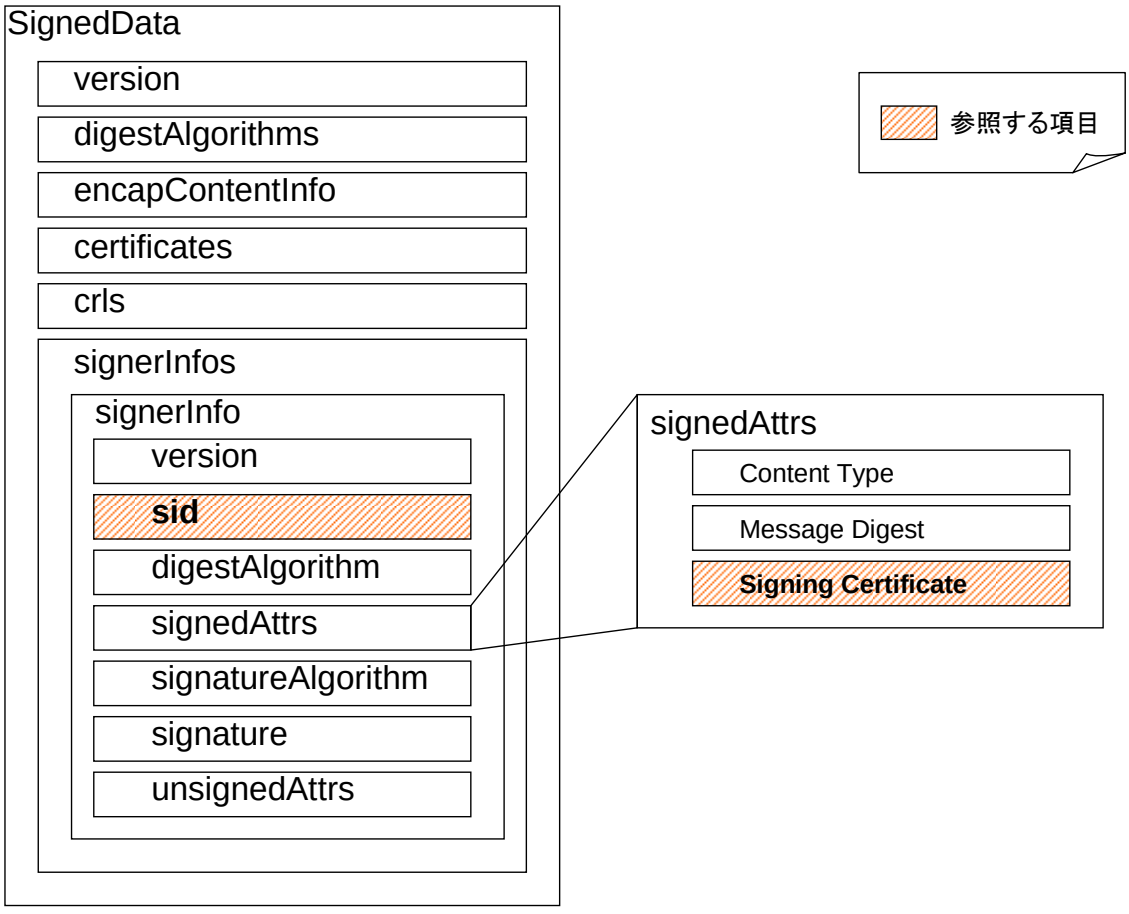


図 2.1.2 署名者証明書の取得と整合性確認で参照する項目

署名時に使用された署名者の証明書と同一のものであることを確認する。
署名者証明書に対して、以下の条件を満たすことを確認する。

a) sid フィールドと一致する かつ b) SigningCertificate 属性と一致する

a) sid フィールドの一致確認
issuerAndSerialNumber フィールドの持つ発行者名とシリアル番号、あるいは subjectKeyIdentifier フィールドの持つ主体者公開鍵識別子が署名者証明書と一致していることを確認する。

b) SigningCerrtificate 属性の一致確認
SigningCertificate 属性のハッシュアルゴリズムに従い、署名者証明書よりハッシュ値を生成し、SigningCertificate 属性に含まれているハッシュ値と一致していることを確認する。さらに、SigningCertificate 属性に issuerSerial が含まれている場合には、発行者の識別名及びシリアル番号が一致していることを確認する。

③　署名値の検証

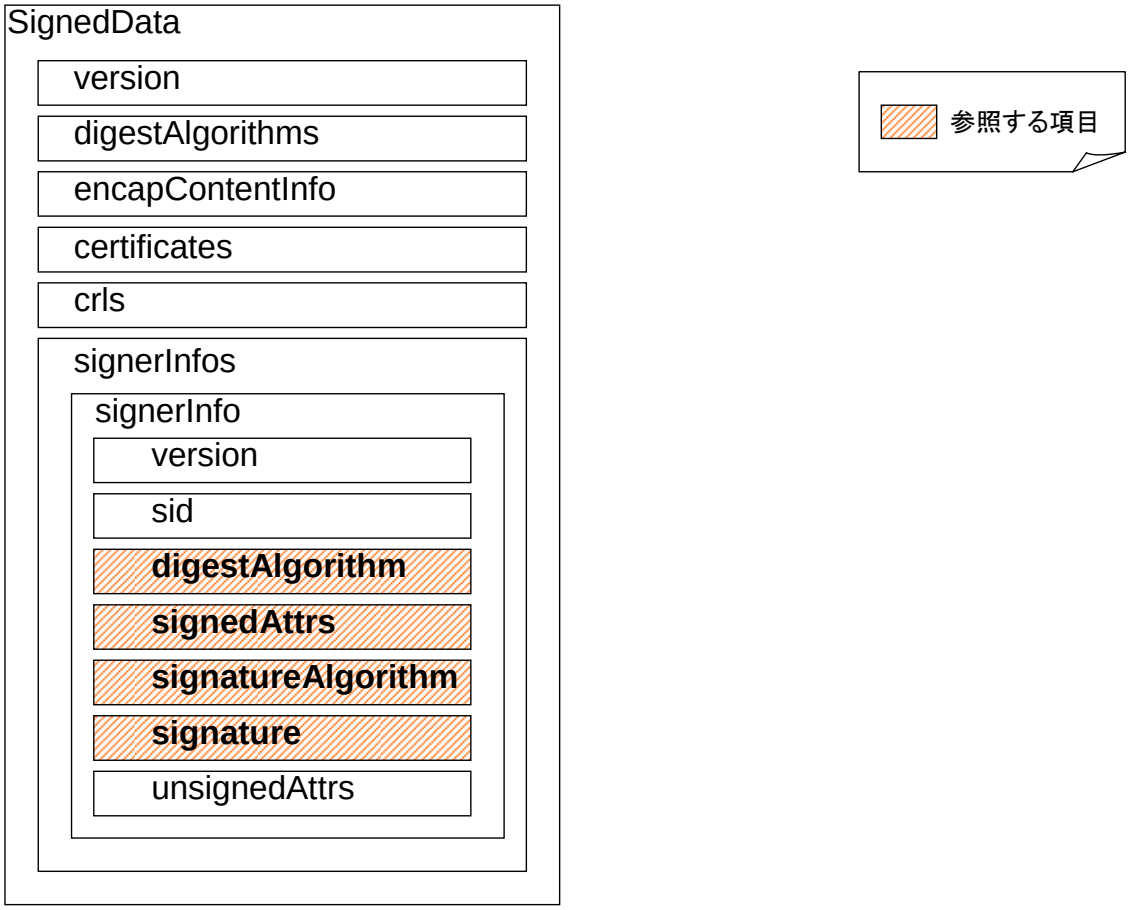


図 2.1.3 署名値の検証で参照する項目

署名者の公開鍵を用いて署名値を検証する。

手順	1. signerInfo より以下の情報を取得する。 ・ signatureAlgorithm フィールドで指定されている署名アルゴリズム ・ digestAlgorithm フィールドで指定されているハッシュアルゴリズム ・ 署名値(signature) ・ 署名属性 signedAttrs 全体の DER データ。ただし、IMPLICIT[0] SET OF タグではなく EXPLICIT SET OF タグとする。 2. 署名属性 signedAttrs 全体の DER データよりハッシュ値を生成する。ハッシュ値生成には digestAlgorithm のハッシュアルゴリズムを用いる。 3. 署名者証明書より取得した公開鍵、署名値、署名属性 signedAttrs のハッシュ値を用いて、署名検証を行う。検証処理の詳細は署名アルゴリズムに従う。

④ 署名者証明書のパス構築・パス検証

トラストアンカとなるルート証明書から署名者証明書までのパス構築・パス検証を行う。1.2 節では検証に必要な証明書や失効情報を取得する方法を、1.3 節は検証で指定する時刻パラメータを各フォーマットのタイプ別にまとめているので参照のこと。

2.2. 署名タイムスタンプの検証

それぞれの signerInfo に含まれる署名タイムスタンプに関して以下の内容を確認する。

a) 署名タイムスタンプのタイムスタンプトークンに含まれる messageImprint フィールドのハッシュ値と signerInfo の signature から導出されたハッシュ値が一致することを確認する。

b) 署名タイムスタンプのタイムスタンプトークンに含まれる signature を TSA 証明書の公開鍵によって検証する。署名値の検証方法は署名者の署名に対する方法（2.1 節③）と同様である。

c) TSA 証明書のパス構築・パス検証を行う。証明書検証に必要な証明書チェーンの取得方法や検証時刻設定については 1.2 節と 1.3 節を参照のこと。また、TSA 証明書の ExtendedKeyUsage 拡張が id-kp-timeStamping であることを確認する。

> RFC 3161 より引用
>
> id-kp-timeStamping OBJECT IDENTIFIER ::= {iso(1) identified-organization(3) dod(6)
>
> internet(1) security(5) mechanisms(5) pkix(7) kp (3) timestamping (8)}

a)から c)のいずれかの検証過程で失敗した場合には、署名タイムスタンプは不正とし検証失敗とする。

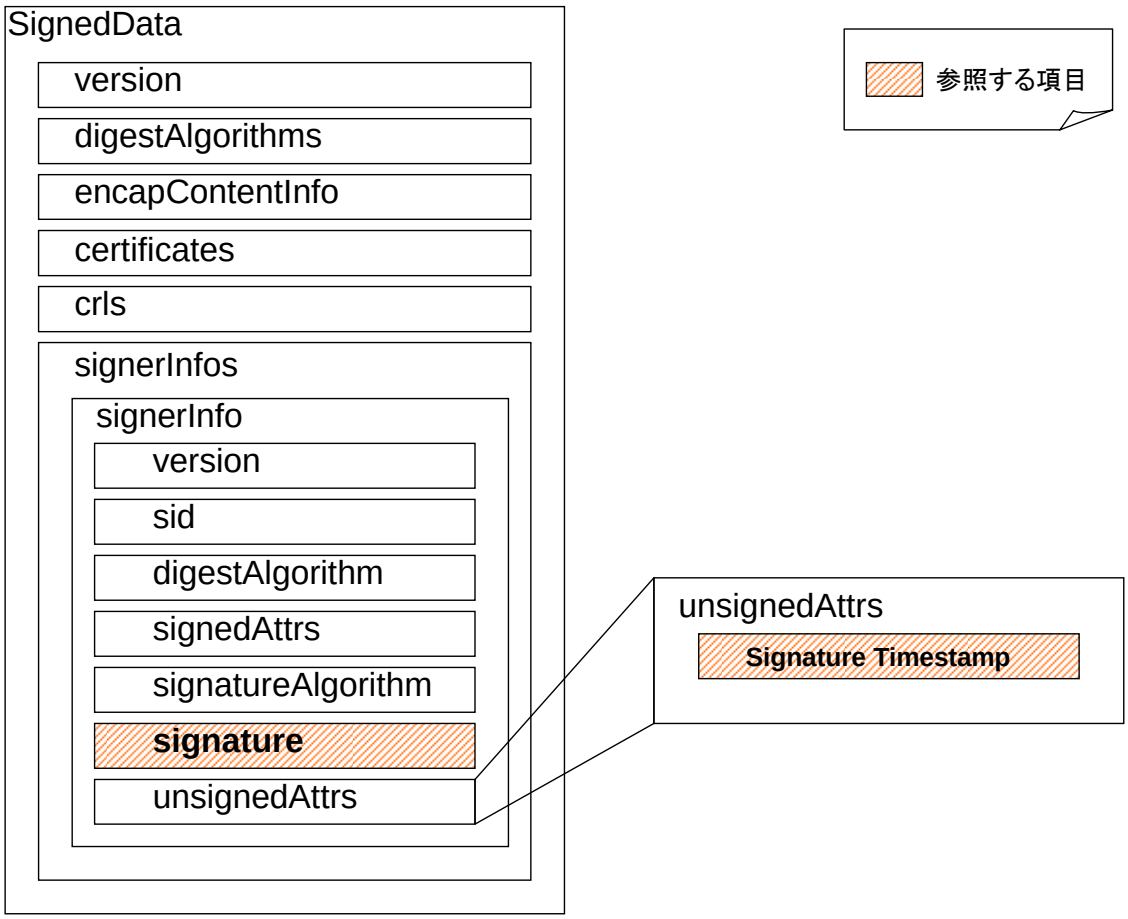


図 2.2.1 署名タイムスタンプ検証で参照する項目

2.3. 証明書・失効情報の参照情報に対する一致確認

- 署名者証明書の証明書チェーン(署名者証明書自身は除く)が非署名属性の CompleteCertificateRefs 属性と一致することを確認する。CompleteCertificateRefs 属性中のそれぞれの OtherCertID に対して
  - 証明書の issuerSerial が一致すること
  - certHash のハッシュ値と証明書から生成したハッシュ値が一致すること

  を確認する。いずれか一つでも OtherCertID が一致しない場合には検証失敗とする。
- 署名者証明書に関する失効情報が非署名属性の CompleteRevocateionRefs 属性と一致するか確認する。それぞれの CrlOcspRef に対して失効情報の識別情報やハッシュ値が一致することを確認する。いずれか一つでも CrlOcspRef が一致しない場合には検証失敗とする。

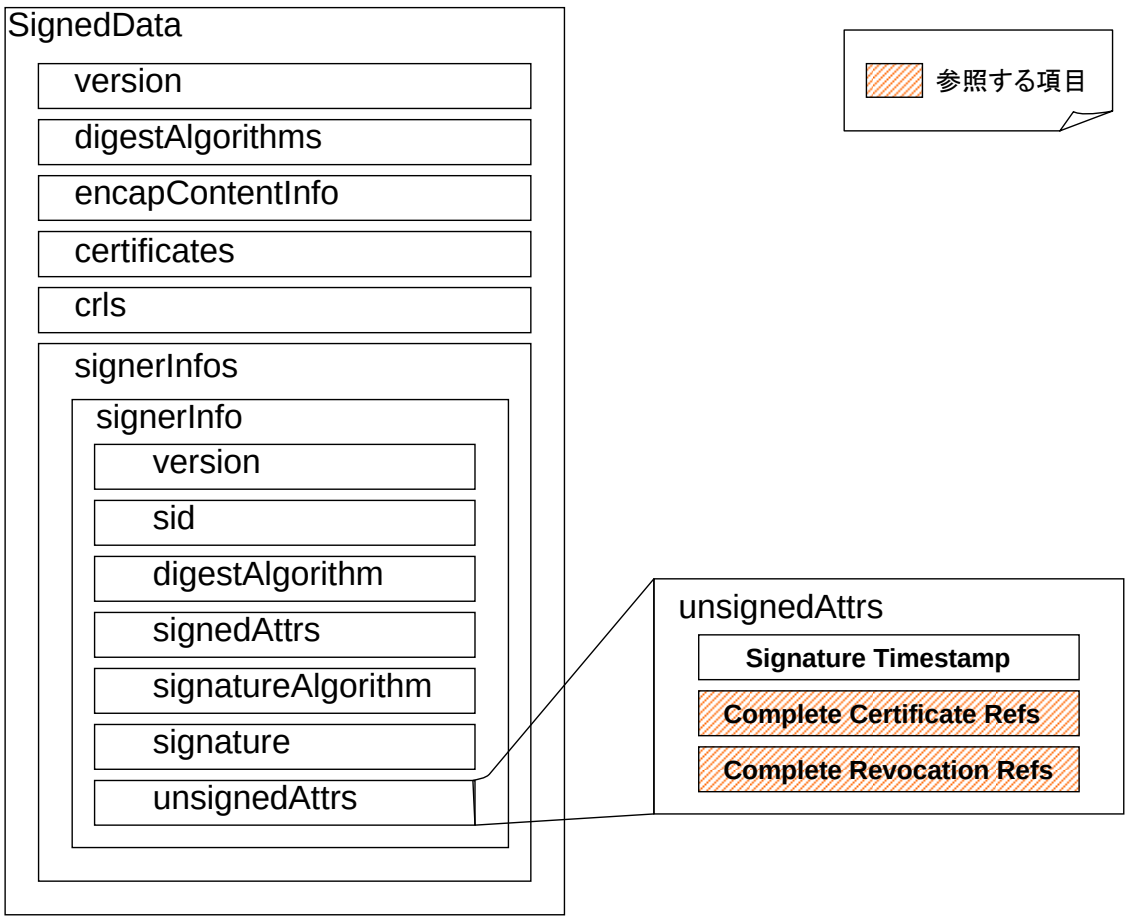


図 2.3.1 証明書・失効情報に対する一致確認で参照する項目

2.4. type1,type2 タイムスタンプの検証

それぞれの signerInfo に含まれる type1,type2 タイムスタンプに関して以下の内容を確認する。

a) タイムスタンプトークンに含まれる messageImprint のハッシュ値と signerInfo の情報から導出されたハッシュ値が一致することを確認する。ハッシュ値の導出方法はタイムスタンプの種類により異なる。ハッシュ対象のまとめは後述する。

b) タイムスタンプトークンに含まれる signature を TSA 証明書の公開鍵によって検証する。署名値の検証方法は署名者の署名に対する方法（2.1 節③）と同様である。

c) TSA 証明書のパス構築・パス検証を行う。証明書検証に必要な証明書チェーンの取得方法や検証時刻の設定については 1.2 節と 1.3 節を参照のこと。また、TSA 証明書の ExtendedKeyUsage 拡張が id-kp-timeStamping であることを確認する。

a)から c)のいずれかの検証過程で失敗した場合には、タイムスタンプは不正とし検証失敗とする。

type1,type2 のハッシュ対象を下記にまとめる。

【type1 タイムスタンプの場合】

CMS 属性の OID	id-aa-ets-escTimeStamp OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 25}
タイムスタンプのハッシュ対象	以下の値(タグと長さを除く)を連結した値 ・ SignerInfo 中の signature フィールドの OCTET STRING ・ SignatureTimeStamp 属性 ・ CompleteCertificateRefs 属性 ・ CompleteRevocationRefs 属性

【type2 タイムスタンプの場合】

CMS 属性の OID	id-aa-ets-certCRLTimestamp OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 26}
タイムスタンプのハッシュ対象	以下の値(タグと長さを除く)を連結した値 ・ CompleteCertificateRefs 属性 ・ CompleteRevocationRefs 属性

2.5. アーカイブタイムスタンプの検証

それぞれの signerInfo に含まれるアーカイブタイムスタンプに対して以下の手順を行う。

i 番目のアーカイブタイムスタンプを検証するものとする。i=1 は最も古いアーカイブタイムスタンプ、i=N を最新のアーカイブタイムスタンプとする。

i 番目(1≦i≦N)のアーカイブタイムスタンプに対して以下を検証する。

a) タイムスタンプ対象の一致確認

【RFC 3126,ETSI TS 101 733 v1.4.0 以前のアーカイブタイムスタンプの場合】

① 1番目から(i-1)番目のアーカイブタイムスタンプに対してアーカイブタイムスタンプ対象のハッシュ生成(messageImprint に相当する値)を行う。以下の値(タイプや長さフィールドを除いた値フィールド)を順に連結した値に対してハッシュ生成する。

* encapContentInfo eContent の OCTET STRING
* signedAttributes
* SignerInfo の signature フィールド
* SignatureTimeStamp 属性
* CompleteCertificateRefs 属性
* CompleteRevocationRefs 属性
* CertificateValues 属性
* RevocationValues 属性
* ESCTimeStampToken 属性（存在する場合）
* TimestampedCertsCRLs 属性（存在する場合）
* 1 番目から(i-1)番目の ArchiveTimeStamp を順に

② i 番目のアーカイブタイムスタンプの messageImprint の値と①で生成したハッシュ値の値が一致することを確認する。

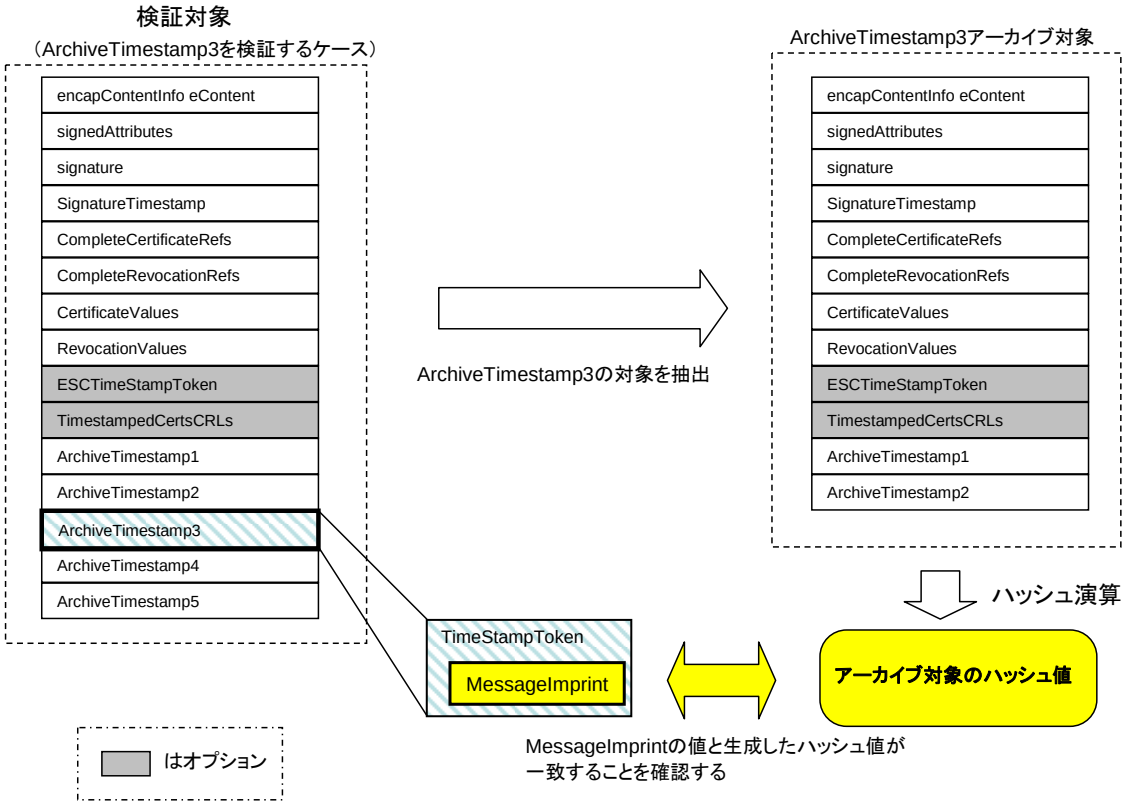


図 2.5.1 RFC 3126 アーカイブタイムスタンプの一致確認

【ETSI TS 101 733 v1.7.3 のアーカイブタイムスタンプの場合】

① 非署名属性から i 番目のアーカイブタイムスタンプを取り除いた非署名属性を新た

に生成する。このとき、以下に注意すること。

- 新たに生成する非署名属性に含まれる各属性の順序が元の非署名属性の配置順序と変わらないこと
- 元のデータのバイナリエンコーディングが変わらないこと

② ①で再生成した非署名属性(unsignedAttrs)を用いて、アーカイブタイムスタンプ対象のハッシュ生成(messageImprint に相当する値)を行う。以下の値（タイプや長さを含んだもの）の順に連結してハッシュ生成を行う。バイナリエンコーディングは元のデータそのままに扱うこと。

* singedData に含まれる encapContentInfo * 外部の署名対象のデータ(encapContentInfo の eContent が省略された場合) * signedData に含まれる certificates と crls フィールド(存在する場合) * signerInfo に含まれる全ての要素（非署名属性は①で再生成したもの).

③ i 番目のアーカイブタイムスタンプの messageImprint の値と③で生成したハッシュ値の値が一致することを確認する。

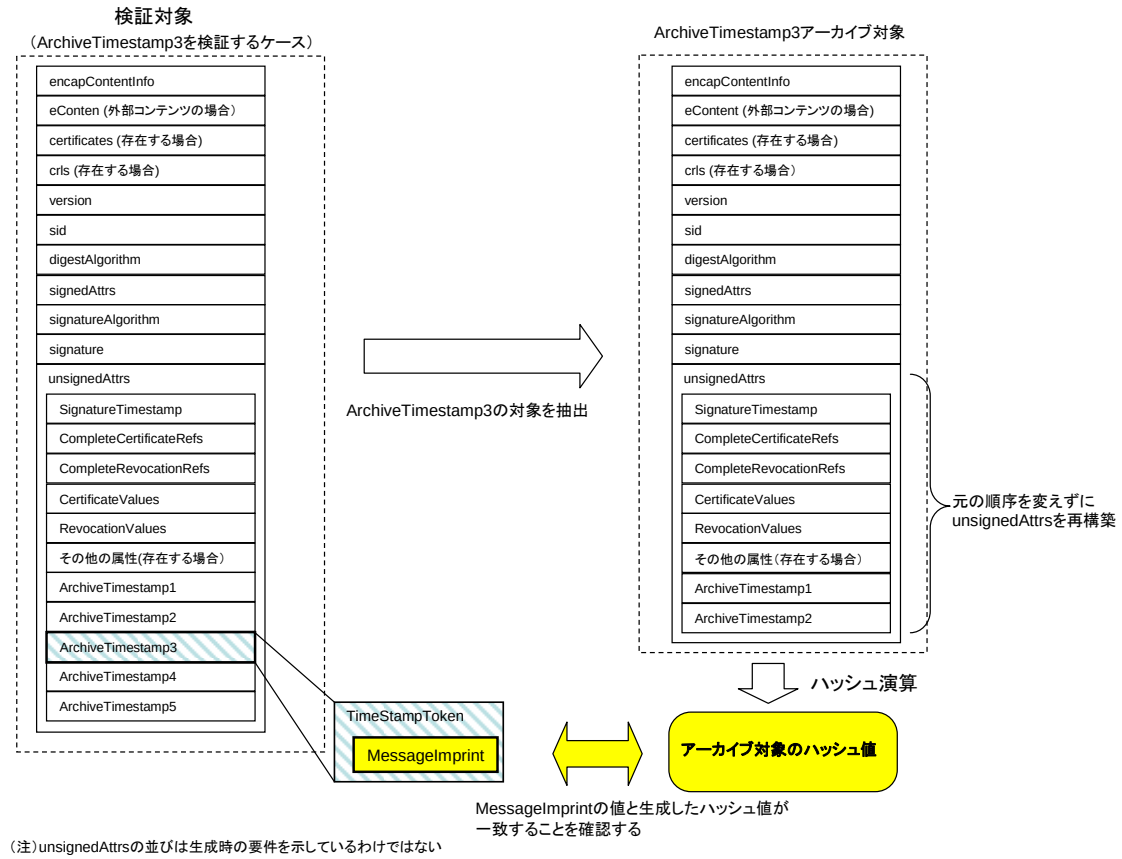


図 2.5.2 ETSI TS 101 733 v1.7.3 アーカイブタイムスタンプの一致確認

b) アーカイブタイムスタンプの署名検証

i 番目のアーカイブタイムスタンプのタイムスタンプトークンに含まれる signature を TSA 証明書の公開鍵によって検証する。署名値の検証方法は署名者の署名に対する方法（2.1 節③）と同様である。

c) TSA 証明書のパス構築・パス検証

トラストアンカとなるルート証明書からエンドエンティティとなる TSA 証明書までのパス構築・パス検証を行う。証明書検証に必要な証明書チェーンが i 番目のアーカイブタイムスタンプに含まれている場合(タイムスタンプトークンの非署名属性)には、それらの情報を用いる。証明書の有効性を確認する時刻設定は(i+1)番目（i=N の場合には検証時点の時刻）を指定する。検証に必要な証明書や失効情報の取得方法や検証時刻設定については 1.2 節と 1.3 節で詳細をまとめているので、合わせて参照のこと。また、TSA 証明書の ExtendedKeyUsage 拡張が id-kp-timeStamping であることを確認する。

a)から c)のいずれかの検証過程で失敗した場合には、アーカイブタイムスタンプは不正とみなし検証失敗とする。

3. 検証処理の詳細（XAdES）

この節では表 1.1.1 に示した各項目の XAdES における詳細について述べる。各フォーマットの検証において該当する検証項目を参照のこと。

3.1. 署名者の署名検証

長期署名フォーマットの以下の項目について確認する。

- 長期署名フォーマットの構造に関する確認
- 署名者証明書の一致確認
- 署名者の署名検証
- 署名者証明書のパス構築・パス検証

3.1.1. 長期署名フォーマットの構造に関する確認

- 長期署名フォーマットの構造に関する確認

  長期署名に関連するすべての要素が ds:Signature 要素の ds:Object 要素に含まれているか、含まれていない場合は QualifyingPropertiesReference で参照されているかを確認する（SHOULD [22] G.2.2.1）。

- SignedProperties 要素に関する確認

  Type 属性に”http://uri.etsi.org/01903#SignedPropertes“という値を持ち、SignedProperties を署名対象とする ds:Reference 要素が存在することを確認する（図 3.1.1.1 参照）。なお、Type 属性に上述の値が指定されていない場合は、XAdES 署名として受け入れるべきではない（SHOULD NOT [22] G.2.2.1）。

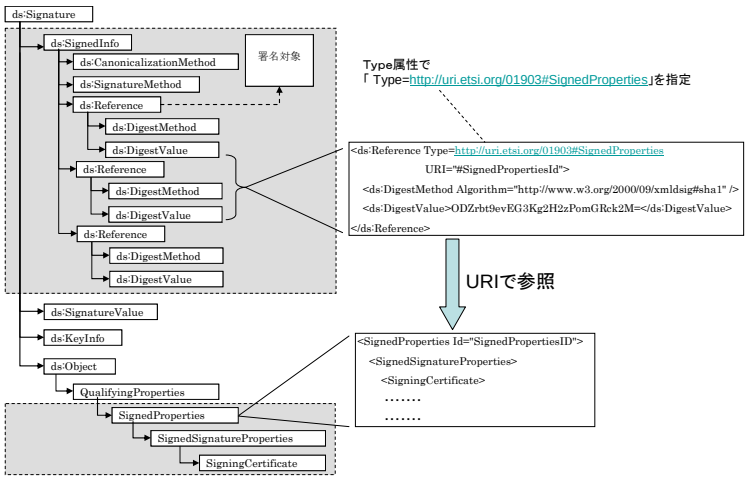


図 3.1.1.1：SignedProperties に関する確認項目

3.1.2. 署名者証明書の一致確認

- 署名者証明書の存在確認

  SigningCerificate 要素の存在を確認する。存在しない場合は、ds:KeyInfo に署名者証明書が含まれるかを確認する。加えて、ds:KeyInfo が署名対象として保護されている（ds:Refernece 要素から参照されている）ことを確認する（SHOULD [22] 4.4.1)。

- 署名者証明書の参照と実体の一致確認

  CertificateValues 要素か、ds:KeyInfo 要素のどちらから署名者証明書の実体を取得する。なお、署名者証明書は上記どちらかの要素に base64 でエンコードされて格納されてり、取り出すにはデコードする必要がある。図 3.1.2.1 に CertificateValues 要素の例および ds:KeyInfo 要素の例を示す。

- 署名者証明書の参照と実体の一致確認

  署名者証明書が特定されたら、SigningCertificate 要素が正しく実体を参照しているかどうかを確認する。なお、適切な署名者証明書への参照が存在しない場合は検証エラーとしなければならない（SHOULD [22] G.2.2.5)。

  1. SigningCertificate 要素内の IssuerSerial 要素に格納されている情報と、署名者証明書の実体の Issuer と serial number が一致しているかを確認する（図 3.1.2.1 の SigningCertificate 要素の例を参照)。

  2. もし、ds:KeyInfo 要素内に ds:X509IssuerSerial 要素が含まれていたら、この要素に格納されている情報と SigningCertificate 要素内の IssuerSerial 要素に格納されている情報が一致していることを確認する（図 3.1.2.1 の ds:KeyInfo 要素の例を参照)。

  3. SigningCertificate 要素内の ds:DigestMethod 要素で指定されたハッシュ関数で計算した署名者証明書のダイジェスト値、ds:DigestValue 要素に格納された値（base64 化されているのでデコードが必要）が一致することを確認する。図 3.1.2.1 に処理概要を図示する。

     ※ SigningCertificate 要素に認証パス中の他の証明書の参照も格納されている場合、それらのチェックも行う。

     ※ SigningCertificate 要素に認証パス構築に関係ない証明書が格納されていた場合は、検証処理は失敗とすべきである(SHOULD NOT)。

※ 署名ポリシで SigningCertificate 要素にすべての認証パスに関する参照情報を格納すべきと定義してあり、SigningCertificate 要素に認証パスの参照情報がそろっていなくても、検証を失敗とすべきではない。

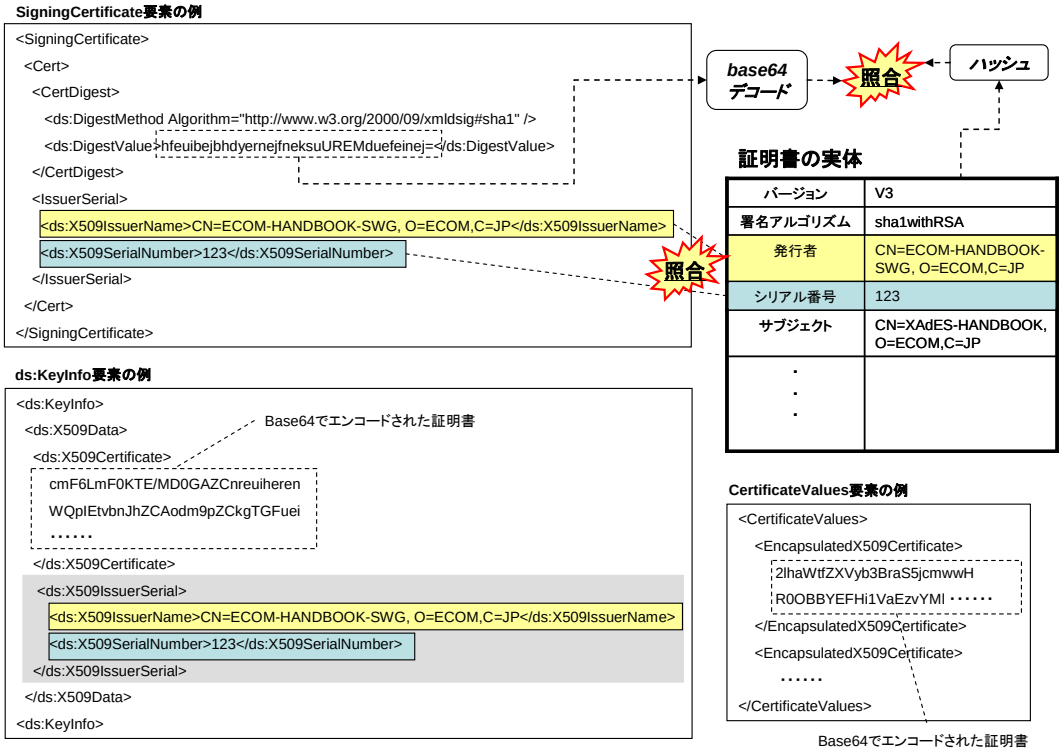


図 3.1.2.1：署名者証明書に関する確認項目

3.1.3. 署名者の署名検証

以下の特定した署名者証明書を用いて以下の表 3.1.3.1 の手順に従い署名の検証を行う。また、検証手順の概要を表 3.1.3.1 に示す。

表 3.1.3.1：署名値の検証手順

手順	1. ds:Reference 要素で参照されている署名対象文書や SignedProperties 要素を取得し以下の手順で改ざんが無いことを確認する。 ① ds:Referenec 要素の配下に変換処理（ds:Transform）が指定されていれば、取得した対象データを変換する。 ② ds:DigestMethod で指定された方法で計算するそのデータのハッシュ値を計算する。 ③ ds:DigestValue に格納されている base64 でエンコードされたハッシュ値を取り出し、デコードする。

	④　デコードした値と②で導いたハッシュ値を照合する。 2. ds:SignedInfo 要素とその配下を取り出す。 3. ds:CanonicalizationMethod で指定されている正規化アルゴリズムで取り出した要素を正規化する。 4. 正規化されたデータのハッシュ値を ds:SignatureMethod で指定されるハッシュ関数でハッシュ計算する（“rsa-sha1”であれば sha1 を用いる）。 5. ds:SignatureValue 要素から base64 でエンコードされた署名値を取り出しデコードする。 6. デコードされた署名値を署名者公開鍵で復号し、４で算出したハッシュ値を照合する。

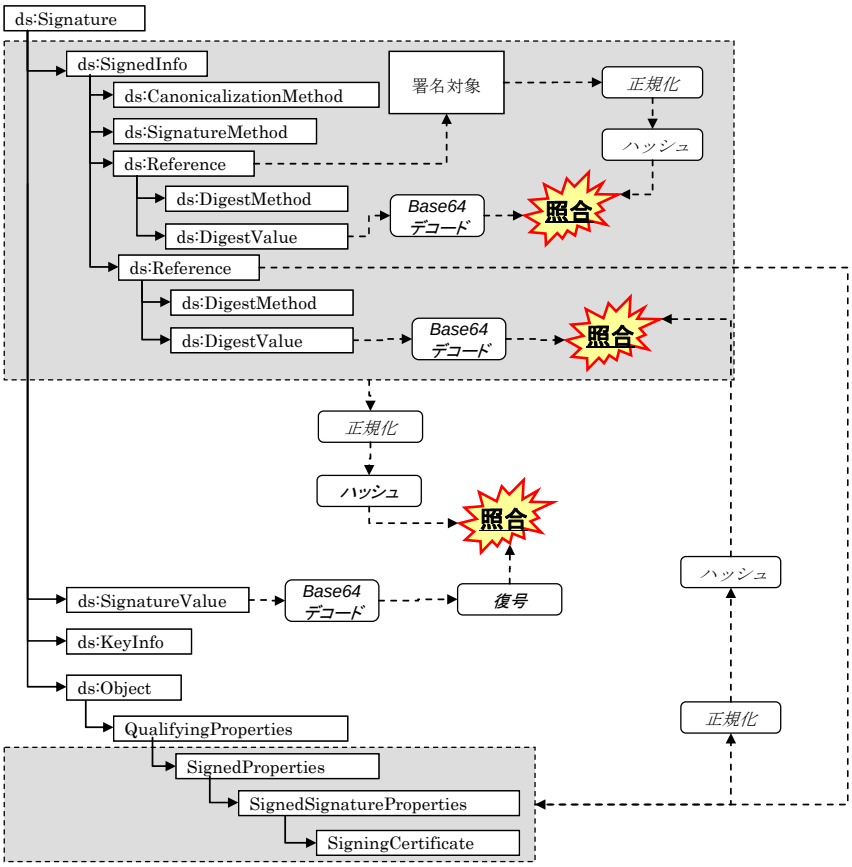


図 3.1.3.1：署名検証時の確認項目

3.1.4.　署名者証明書のパス構築・パス検証

トラストアンカとなるルート証明書から署名者証明書までのパス構築･パス検証を行う。

認証パスを構築に必要な証明書は、CertificateValues 要素か ds:KeyInfo 要素から取り出す。また、検証情報は RevocationValues 要素から取り出す。なお、そのとき以下の点も確認する必要がある。

- CertificateValues と ds:KeyInfo に格納されている証明書で認証パスが構築できるかどうかを確認する。もし、認証パスが構築できなければ検証処理は失敗とすべきである。

- RevocationValues が存在する場合は、RevocationValues に認証パス検証に必要なすべての検証情報が確認されている必要がある。

パス構築・パス検証においてはトラストアンカとなるルート証明書を指定する必要があるが、このトラストアンカとなる証明書は長期署名フォーマットのデータとは別に安全に管理する必要がある。1.2 章では検証に必要な証明書や失効情報を取得する方法を、1.3 章では検証で指定する時刻パラメータを各フォーマットのタイプ別にまとめているので参照のこと。

3.2. 署名タイムスタンプの検証

署名タイムスタンプに関しては以下の手順で検証処理を行い正しいことを確認する。また、処理概要を図 3.2.1 に示す。

1. タイムスタンプトークンの取り出し
   SignatureTimeStamp 要素の EncapsulatedTimeStamp 要素から base64 でエンコードされたタイムスタンプトークンを取り出しデコードする。

2. タイムスタンプトークンの検証
   タイムスタンプトークンの署名検証を行う。タイムスタンプトークン自体の検証に関する規定は ETSI TS 101 903 V1.3.2([22])では規定範囲外となっているが、ここでは RFC 3161[5]タイムスタンプの検証方法について述べる。RFC 3161[5]とは異なるタイムスタンプを使用する場合には、別途規定された方法に従い検証を行うこと。

   - タイムスタンプトークンに含まれる signature を TSA 証明書の公開鍵によって検証する。RFC 3161[5]タイムスタンプの署名値の検証方法は CAdES の署名者の署名に対する方法（2.1 節③）と同様である。

   - TSA 証明書のパス構築・パス検証を行う。証明書検証に必要な証明書チェーンの取得方法や検証時刻設定については、1.2 章、1.3 章を参照のこと。

   上記の処理の過程で失敗した場合には、署名タイムスタンプは不正とし検証失敗とする。

3. ds:SignatureValue 要素を取り出しと正規化

   ds:SignatureValue を取り出し、取り出したものに対して正規化を行う。正規化には、もし SignatureTimeStamp 要素内に CanonicalizationMethod 要素が存在すればそれに従った正規化を、存在しなければ XML 署名規格における標準的な方法で正規化を行う。

4. ハッシュ値の計算と突合せ

   タイムスタンプトークン内に記述されているハッシュ関数で、正規化された ds:SignatureValue のハッシュを計算する。その結果とタイムスタンプトークン内に格納されたタイムスタンプ対象の値（ハッシュ値）を比較する。

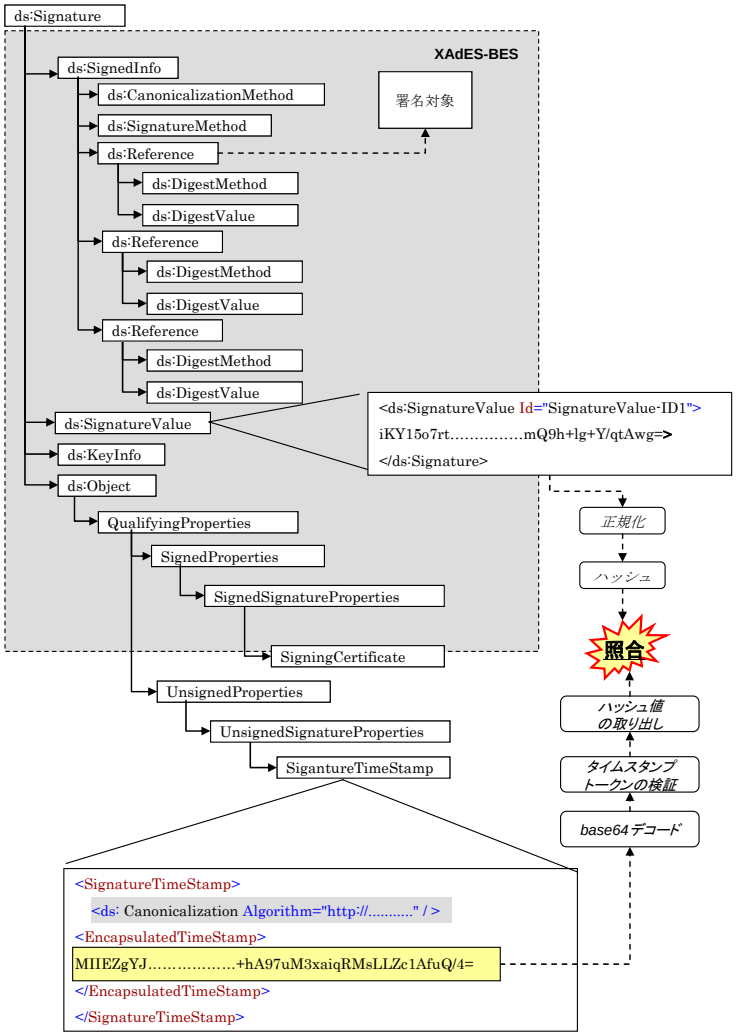


図 3.2.1：署名タイムスタンプに関する確認項目

3.3. 証明書・失効情報の参照情報に対する一致確認

- 署名者証明書の証明書チェーン（署名者証明書自身は除く）が UnsignedSignatureProperties 要素の CompleteCertificateRefs 要素の内の証明書に対する参照情報(CertRefs 要素内の Cert 要素)と一致することを確認する。
  1. Cert 要素内の IssuerSerial 要素に格納されている情報と、署名者証明書の実体の Issuer と serial number が一致しているかを確認する（図 3.3.1 の CompleteCertificateRefs 要素の例を参照）。
  2. Cert 要素内の ds:DigestMethod 要素で指定されたハッシュ関数で計算した各証明書のダイジェスト値と、ds:DigestValue 要素に格納された値（base64 化されているのでデコードが必要）が一致することを確認する。図 3.3.1 に処理概要を図示する。

- 署名者証明書に関する失効情報が UnsignedSignatureProperties 要素の CompleteRevocationRefs 要素の内の失効情報に対する参照情報(CRLRefs 要素内の CRLRef 要素や OCSPRefs 要素内の OCSPRef 要素)と一致することを確認する。ここでは失効情報が CRL の場合を例にとり説明する（図 3.3.1 の CompleteRevocationRefs 要素の例を参照）。
  1. CRLRef 要素内の CRLIdentifier 要素内の Issuer 要素に格納されている情報と、CRL の実体の Issuer が一致しているかを確認する
  2. CRLRef 要素内の CRLIdentifier 要素内の IssueTime 要素に格納されている情報と、CRL の実体の thisUpdate が一致しているかを確認する。
  3. CRLRef 要素内の CRLIdentifier 要素内の Number 要素に格納されている情報と、CRL の実体の cRLNumber が一致しているかを確認する。
  4. CRLRef 要素内の ds:DigestMethod 要素で指定されたハッシュ関数で計算した各 CRL のダイジェスト値と、ds:DigestValue 要素に格納された値（base64 化されているのでデコードが必要）が一致することを確認する。

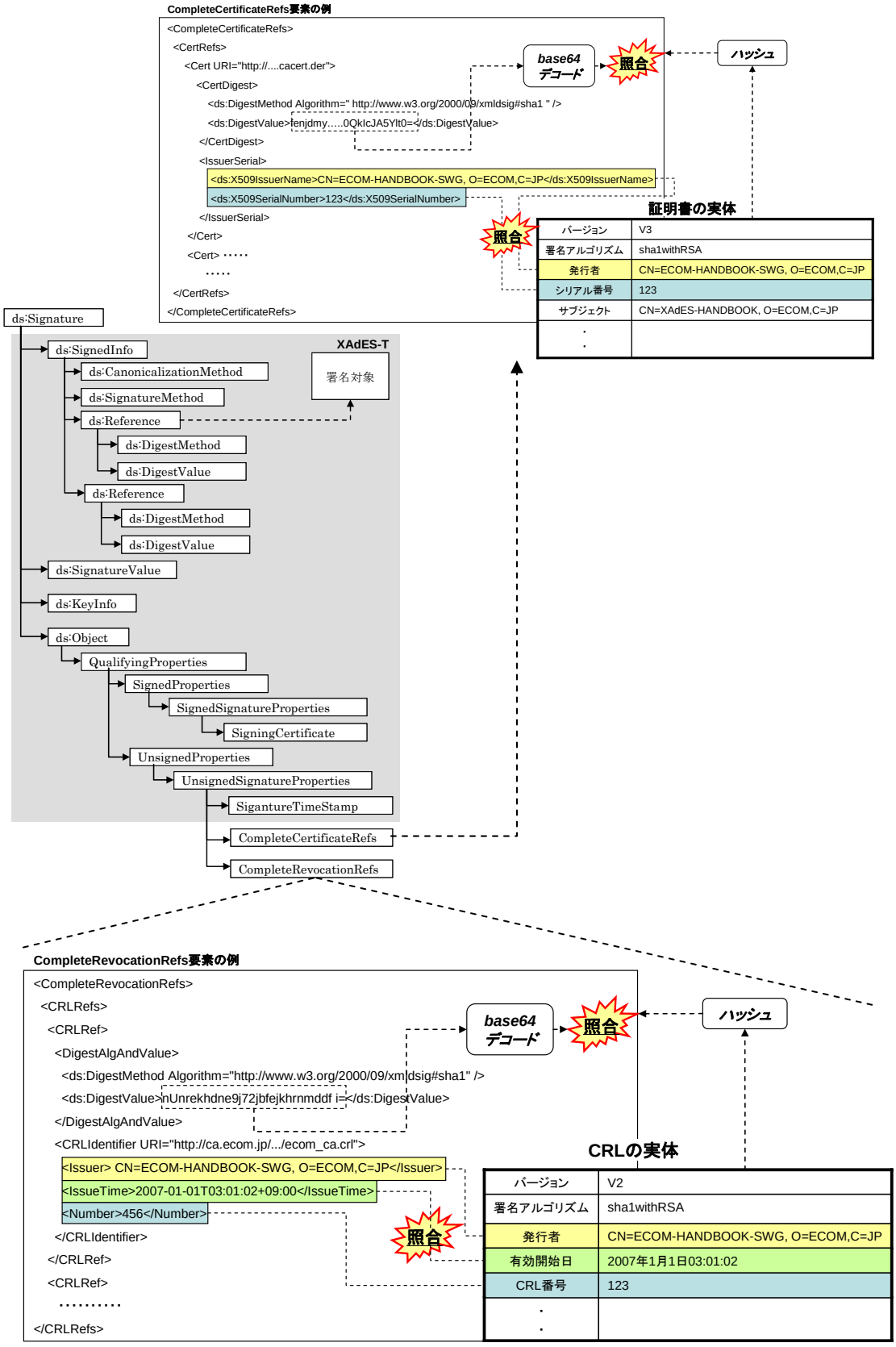


図 3.3.1：証明書・失効情報の参照情報に対する一致確認における確認項目

3.4. 失効情報を保護するタイムスタンプトークンの検証

失効情報を保護するタイムスタンプトークンは、署名値を保護対象とするかどうかで RefOnlyTimeStamp と SigAndRefsTimeStamp の 2 種類が定義されている。ここでは、RefsOnlyTimeStamp の not distributed case を例として検証手順を説明する。図 3.4.1 に検証・確認手順の概要を図示し、以下に手順を説明する。

1. 検証対象タイムスタンプトークンの取り出し

   RefsOnlyTimeStamp 要素（または SigAndRefsTimeStamp 要素）の EncapsulatedTimeStamp 要素から base64 でエンコードされたタイムスタンプトークンを取り出しデコードする。

2. タイムスタンプトークンの検証

   タイムスタンプトークン自体の検証に関する規定は ETSI TS 101 903 V1.3.2[22]では規定範囲外となっているが、ここでは RFC 3161[5]タイムスタンプの検証方法について述べる。RFC 3161[5]とは異なるタイムスタンプを使用する場合には、別途規定された方法に従い検証を行うこと。

   - タイムスタンプトークンに含まれる signature を TSA 証明書の公開鍵によって検証する。RFC 3161[5]タイムスタンプの署名値の検証方法は CAdES の署名者の署名に対する方法（2.1 節③）と同様である。
   - TSA 証明書のパス構築・パス検証を行う。証明書検証に必要な証明書チェーンの取得方法や検証時刻設定については、1.2 章、1.3 章を参照のこと。

   上記の処理の過程で失敗した場合には、アーカイブタイムスタンプは不正とし検証失敗とする。

3. タイムスタンプの時刻の確認
   タイムスタンプの時刻が SigningTime や SignatureTimeStamp（AllDataObjectsTimeStamp や IndividualDataObjectsTimeStamp が存在すればそれよりも)、より後の時刻であること、及び ArchiveTimeStamp が存在すればその時刻より前であることを確認する。

4. ハッシュ値の計算と突合せ

   - タイムスタンプの付与対象である CompleteCertificateRefs 要素および CompleteRevocationRefs 要素を取り出す（SigAndRefsTimestamp の場合は、ds:SignatureValue 要素、SignatureTimeStamp 要素も取り出す)。
   - 取り出した要素を正規化する。正規化には、もし RefsOnlyTimeStamp(または、SigAndRefsTimeStamp)要素内に CanonicalizationMethod 要素が存在すれば

それに従った正規化を、存在しなければ XML 署名規格における標準的な方法で正規化を行う。

- UnsignedSignaturePropeties 要素内の順番に従って連結する。
- タイムスタンプトークン内に記述されているハッシュ関数で、連結したデータのハッシュ値を計算する。その結果とタイムスタンプトークン内に格納されたタイムスタンプ対象の値（ハッシュ値）を比較する。

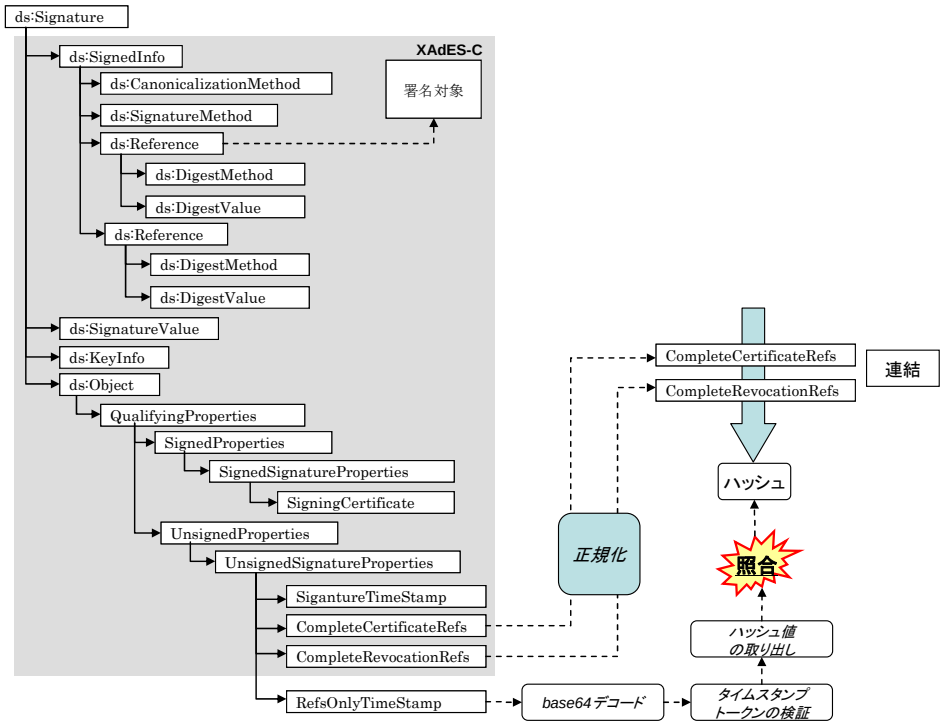


図 3.4.1：失効情報を保護するタイムスタンプトークンに関する確認項目

3.5. アーカイブタイムスタンプの検証

アーカイブタイムスタンプは以下の手順で検証する。ここでは、not distributed case について説明する。検証は以下の手順を実行する。また、図 3.5.1 に検証手順の概要を図示する。ここでは、i=1 を最も古いアーカイブタイムスタンプ、i=N を最新のアーカイブタイムスタンプとしたときの i 番目（1≦i≦N）のアーカイブタイムスタンプを検証するものとして説明する。

1. 検証対象タイムスタンプトークン（i 番目）の取り出し

   ArchiveTimeStamp 要素の EncapsulatedTimeStamp 要素から base64 でエンコードされたタイムスタンプトークンを取り出しデコードする。

2. タイムスタンプトークンの検証

タイムスタンプトークン自体の検証に関する規定は ETSI TS 101 903 V1.3.2[22]では規定範囲外となっているが、ここでは RFC 3161[5]タイムスタンプの検証方法について述べる。RFC 3161[5]とは異なるタイムスタンプを使用する場合には、別途規定された方法に従い検証を行うこと。

- タイムスタンプトークンに含まれる signature を TSA 証明書の公開鍵によって検証する。RFC 3161[5]タイムスタンプの署名値の検証方法は CAdES の署名者の署名に対する方法（1.1.8 節③）と同様である。
- TSA 証明書のパス構築・パス検証を行う。証明書検証に必要な証明書チェーンの取得方法や検証時刻設定については、1.2 章、1.3 章を参照のこと。

上記の処理の過程で失敗した場合には、アーカイブタイムスタンプは不正とし検証失敗とする。

3. 署名対象要素に関する処理。
以下の順序で要素を取り出す。

- ds:Reference 要素（ds:SignedInfo 内の出現順序に従い取り出す）
- ds:SignedInfo 要素
- ds:SignatureValue 要素
- ds:KeyInfo 要素（存在する場合のみ）

取り出した ds:Reference 要素を XML 署名規格で定義された方法で処理する。処理結果と正規化・連結し octet stream を生成する。残りの要素も正規化しおよび連結し最終的な octet stream を生成する。

4. 署名対象ではない要素に関する処理
CertificateValues 要素と RevocationValues 要素が存在することを確認する。もし、どちらか一方でも存在しない場合は、検証失敗とする。次の要素の存在を確認し、以下の順序で要素を取り出す。取り出した要素を正規化し、その結果を 3.で生成した octet stream と連結する

- SignatureTimeStamp 要素
- CounterSignatrue 要素（存在した場合のみ）
- CompleteCertificateRefs 要素（存在した場合のみ）
- CompleteRevocationRefs 要素（存在した場合のみ）
- AttributeCertificateRefs 要素（存在した場合のみ）
- AttributeRevocationRefs 要素（存在した場合のみ）
- CertificateValues 要素
- RevocationValues 要素
- SigAndRefsTimeStamp 要素（存在した場合のみ）
- RefsOnlyTimeStamp 要素（存在した場合のみ）
- ArchiveTimeStamp 要素（1 番目から i-1 番目）
- ds:Object 要素（ds:Referece 要素で参照されていないものが存在すれば、それらすべて）

5. タイムスタンプトークンの中に記述されたハッシュ関数に従い、上記結果バイト列に対するハッシュ値を計算する。計算結果のハッシュ値とタイムスタンプトークンに格納されているハッシュ値を付き合わせる。

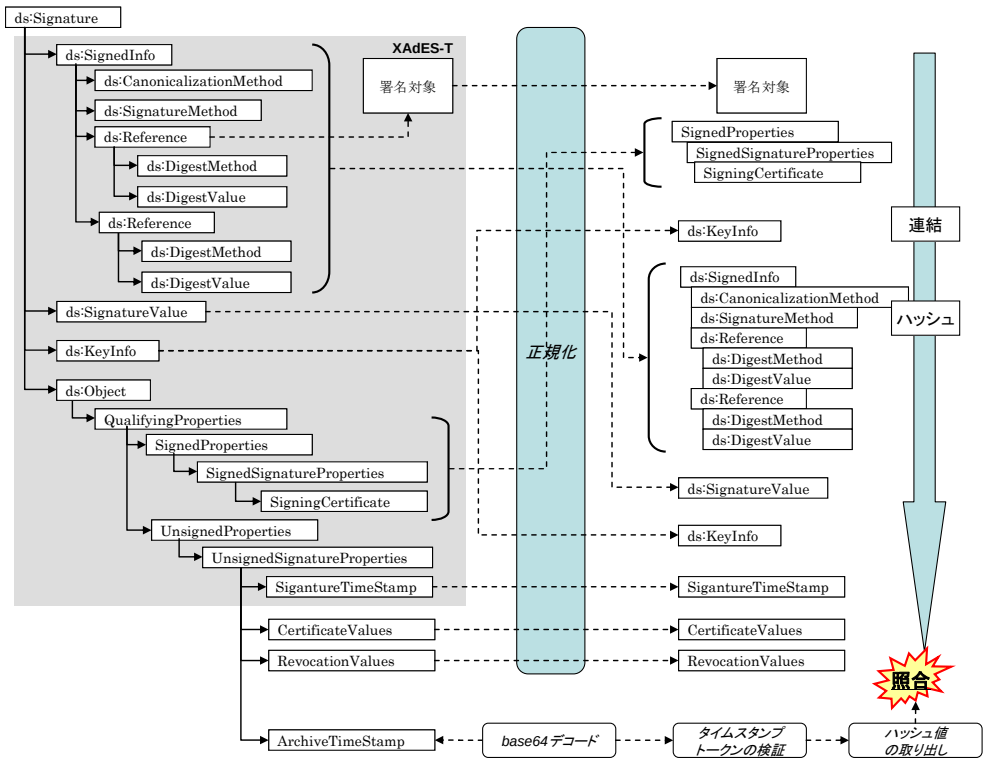


図 3.5.1：アーカイブタイムスタンプに関する確認項目

第一部　デジタル署名ハンドブック

付録

1. 付録 1：PKCS#7 と CMS の SignedData の違い

CAdES は、CMS SignedData[10]に基づく署名データフォーマットである。CMS は PKCS#7[30]を拡張したセキュリティデータフォーマットの国際標準であり、PKCS#7 との後方互換性を備えている。では、PKCS#7 を CAdES のための署名データのベースとして使うことができるだろうか。これに対する簡潔な答えは「CAdES v1.7.3 以降ならば PKCS#7 ベースでも CAdES フォーマットとすることができる」ということである。

CAdES v1.7.3 より前では、SignedData のバージョンが 3 である必要があったため、PKCS#7 とすることができなかったが、CAdES v1.7.3 では、このような制限が無くなったために PKCS #7 をベースとした CAdES を利用できるようになった。この事により、PKCS#7 しか許されていない PDF 署名において、PKCS#7 による CAdES-T 署名タイムスタンプを含めることができるようになった。

本節では、PKCS#7 と CMS の SignedData について、その違いを説明する。

1.1. PKCS#7 と CMS の利用

PKCS#7 や CMS は、署名や暗号に関連する様々な構造化データを交換する際に利用されている。両者の用途や仕様策定について違いをまとめたのが以下である。

	PKCS#7	CMS
用途	暗号，署名，S/MIME，証明書セット, PDF 署名など	暗号，署名，S/MIME，CAdES，タイムスタンプトークンなど
説明	古くからあるセキュリティデータに関する仕様であり、既に広く利用されている RSA Laboratories が策定した業界標準。仕様が長く更新されていないため、CAdES など新しい標準と紐付けできない場合がある。古く策定されたデータ構造は、依然として利用されている。	PKCS#7 を元に IETF が策定した国際標準。PKCS#7 と後方互換性を持つ。新しい標準では PKCS#7 よりも CMS が使われている。仕様が数年毎に更新されている。
最新版	PKCS#7 1.5 (1993 年 11 月公開)	RFC 3852 (2004 年 6 月公開)

PDF 署名について、これまで ECOM では、ISO 並びに米 Adobe 社など PDF/A において CAdES/XAdES 長期署名フォーマットを利用できるようにするために提言、連携並びに調整を行っている。

また、S/MIME 署名メールについては、現在流通している多くの実装が、署名による改ざん対策よりも相互運用性を比重に置いているため、古くから利用されてきた PKCS#7 に基づいており、CMS に基づく CAdES を S/MIME 署名に利用できない場合があるかもしれないため、CMS の後方互換性に注意しながら利用する必要がある。

1.2. PKCS#7 SignedData と CMS SignedData の違い

PKCS#7 と CMS の署名データ構造である SignedData の違いを重要なものから順に以下にまとめる。

- SignedData 中の署名対象文書の情報を保持する contentInfo フィールドの違い。PKCS#7 では contentInfo, CMS では encapContentInfo。
- SignerInfo の sid で PKCS#7 では issuerAndSerialNumber のみ
- PKCS#7 では SignedData のバージョンは 1 のみ(SHALL[30]9.1)
- PKCS#7 では certificates フィールドには属性証明書は入らない
- SignerInfo の署名アルゴリズム、署名値フィールドの名称。PKCS#7 は digestEncryptionAlgorithm と encryptedDigest、CMS は signatureAlgorithm と signature となる。

以降、重要な点について詳しく解説する。

1.1.1. PKCS#7/contentInfo と CMS/encapContentInfo の違い

署名対象文書やデータの情報を保持するために、PKCS#7 では contentInfo フィールド、CMS では encapContentInfo フィールドが使用される。

PKCS#7[30]での contentInfo の定義は、CMS 全体を構成する ContentInfo の定義と同じものを SignedData の中でも使用している。署名対象文書の格納方法について制限の非常に緩いものとなっている。content の定義は以下の通り。

```
content [0] EXPLICIT ANY DEFINED BY contentType OPTIONAL
引用：PKCS #7 v1.5 [30]
```

一方、CMS では SignedData で署名対象文書を含める場合には汎用の contentInfo ではなく、encapContentInfo という content を ASN.1 OCTET STRING 型でカプセル化したデータ構造としている。

eContent [0] EXPLICIT OCTET STRING OPTIONAL 引用：RFC 3852

これに従い、CMS では Microsoft Word や一太郎、ASN.1 構造を持つデータなど任意のデータを OCTET STRING にカプセル化した状態で持たせることができるが、PKCS#7 の場合には、直接 ASN.1 構造を埋める場合には何らかの規定が必要になる場合がある。署名対象を一般的な data コンテントタイプとする場合には、PKCS#7 であっても OCTET STRING でカプセル化することになる。CMS と PKCS#7 の互換性については、RFC 3852[10]の 5.2.1 節でも述べられているので参考にされたい。

1.1.2. PKCS#7/encryptedDigest と CMS/signature の違い

最初に、署名者各々の署名情報を保持する SignedData の SignerInfo 構造の PKCS#7、CMS 両者の ASN.1 定義を比較のため以下に示す。太字で示されたところ名称が違う箇所である。

PKCS#7 1.5[10] SignedData SignerInfo	RFC 3852[30] CMS SignedData SignerInfo
SignerInfo ::= SEQUENCE { version Version **issuerAndSerialNumber** **IssuerAndSerialNumber,** digestAlgorithm DigestAlgorithmIdentifier, authenticatedAttributes [0] IMPLICIT Attributes OPTIONAL, **digestEncryptionAlgorithm** ハッシュ暗号化アルゴリズム **DigestEncryptionAlgorithmIdentifier,** **encryptedDigest** 暗号化されたハッシュ値 **EncryptedDigest,** unauthenticatedAttributes [1] IMPLICIT Attributes OPTIONAL } 引用：PKCS#7 1.5[10] 9.2 節より	SignerInfo ::= SEQUENCE { version CMSVersion **sid** **SignerIdentifier,** digestAlgorithm DigestAlgorithmIdentifier, signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL, **signatureAlgorithm** 署名アルゴリズム **SignatureAlgorithmIdentifier,** **signature** 署名値 **SignatureValue,** unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL } 引用：RFC 3852[30] 5.3 節より

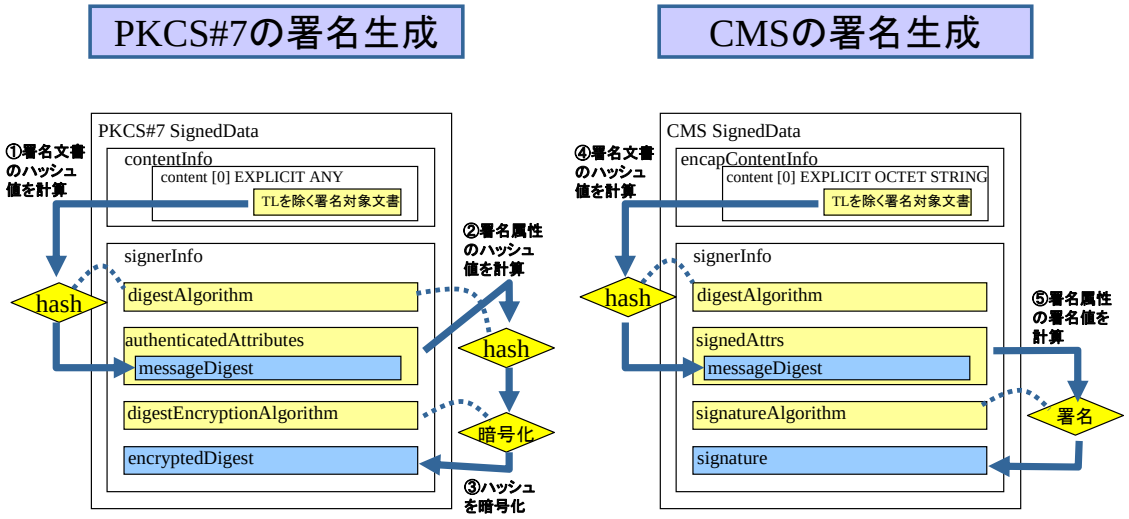
digestEncryptionAlgorithm と signatureAlgorithm とで、利用可能なアルゴリズムをまとめると以下のようになる。

digestEncryptionAlgorithm の値	signatureAlgorithm の値
rsaEncryption md5WithRSA sha1WithRSA (PKCS #7 [30] より)	rsaEncryption (MUST) md5WithRSA (MAY) sha1WithRSA (MAY) sha1WithDSA (RFC 3370[8] 3 より)

ここで、rsaEncryption とあるが、CMS で用いられる暗号アルゴリズムについて規定した RFC 3370[8] 3 節で述べられているように、これは暗号アルゴリズムを指しているのではなく、メッセージダイジェストアルゴリズムを特定しない任意の RSA 暗号を用いた署名アルゴリズムを指している。

2006 年後半、IETF S/MIME ワーキンググループメーリングリストでは、signatureAlgorithm と SignerInfo の digestAlgorithm に関連があるのか、無いのかが議論された。現時点での仕様では、signatureAlgorithm と digestAlgorithm には関連があるとは記述されていない。この問題は CAdES の場合には大きな影響がある。

CAdES においては署名属性群(authenticatedAttributes や signedAttrs)は必須であるが、その場合、現時点の仕様からは digestAlgorithm と signatureAlgorithm を独立とすることもできるのである。例として署名者属性群があり内包署名である時に、digestAlgorithm と signatureAlgorithm が独立である場合の署名生成の違いについて図を用いて説明する。



PKCS#7 SignedData の場合には、左図の処理①と処理②のハッシュアルゴリズムは同じでなければならないため、例えば digestAlgorithm に RIPEMD-160 のような国内ではあまり一般的でないハッシュアルゴリズムを用いたとしたら、処理②もこれに従うこととなる。一方、CMS の場合には、digestAlgorithm と signatureAlgorithm は独立であるとした場合、例えばハッシュに RIPEMD-160 を使い、署名に sha1WithRSA を使うことが可能である。相互運用性や CMS の今後の改定を想定すると、digestAlgorithm と signatureAlgorithm は関係があるとして、実装した方が良いと考えられる。

昨今の暗号モジュールやライブラリでは、否認防止を目的として署名専用の鍵ペアを用意するという考え方から純粋に公開鍵暗号の私有鍵による暗号化処理を単体で行うことができないモジュールが多い。従って PKCS#7 であったとしても処理②と処理③をまとめて署名処理として行うことになる。このような暗号モジュールを使用して PKCS#7 SignedData を生成する場合、digiestAlgorithm は署名アルゴリズムとしてサポートされているものしか選択できないことに注意する必要がある。

2. 付録 2：複数署名

現実社会では、複数の署名者が同一の署名対象文書に対して署名を行うようなケースがある。例えば、契約や法的要件に基づき当事者全員の署名が求められたり、担当者が上申し所属長が承認を行ったり、また契約など 2 つの組織が対等な立場で署名対象文書に対し承認する意味で署名を行うことがある。CMS SignedData [10] や XML 署名 [32] では、このようなケースでも利用できるように複数署名(multiple signatures)と呼んでいる複数の署名者による署名をサポートしている。

複数の署名を行う場合、前述の例で述べたように、署名の順序が問題となる場合と、順序は関係なく独立して一つの署名対象文書に対して署名を行う場合がある。前者をカウンタ署名(counter signatures)、後者を独立署名(independent signatures)と呼んでいる。

CAdES や XAdES の標準では、CAdES v1.7.3[18] 5.12 節 Support for multiple signatures、ならびに XAdES v1.3.2[22] Appendix C.1 節 (Informative) Multiple signatures and countersignatures において、複数署名について簡単に触れられているが、長期保存に必要となるタイムスタンプや検証情報の格納方法について詳しく述べられていないため、本節で解説することとする。

2.1. 複数署名の用語の整理

複数署名には幾つかの形態があり、これらを様々な呼称、例えば並列署名、直列署名、組み込み署名などと呼んできた。本節では複数署名の形態および呼称をまず整理することとする。

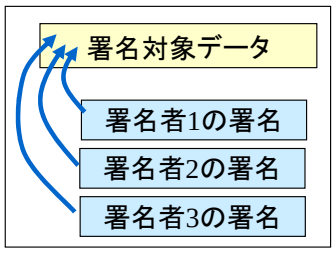
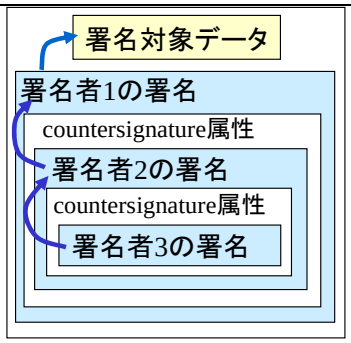
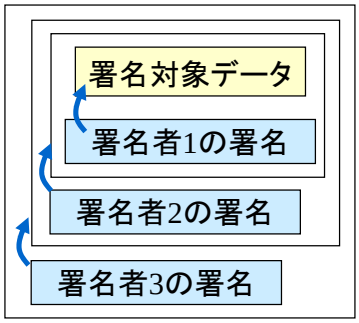
呼称	独立署名 independent signature 並列署名 parallel signature	組み込み署名 embedded signature カウンタ署名 / 対向署名 counter signature 直列署名 serial signature	組み込み署名 embedding signature 全体署名 overall signature 直列署名 serial signature
順序	署名の順序関係がない	署名の順序関係がある	署名の順序関係がある
形態	一つの署名対象に対して複数の署名者が独立に行う署名	署名対象となる署名データ(構造)の中に、署名を追加する人の署名を入れる。	署名対象となる署名データは、署名を追加する人の署名(構造)の中に入れられる。
標準	CMS、XML 署名、共に対応。	CMS、XML 署名、共に対応。	XML 署名では標準として対応。CMS の場合にはアプリケーションとして実現でき仕様の範囲外。
長所・短所		カウンタ署名を処理できない実装でも、汎用の機能で最初の署名者の署名を取り出せる。	CMS の場合には、最初の署名者の署名の取り出し方をアプリケーションで規定する必要があり面倒。
データ構造図		 (内包署名の場合)	 (内包署名の場合)

図 2.1.1: 複数署名の分類

2.2. 複数署名が独立署名のみの場合

独立署名のみである場合には、署名データ構造に署名を複数持たせることにより、複数人が独立して行う署名を表現することができる。CMS 署名の場合には、SignerInfo を複数も持たせることにより、また XML 署名の場合には外部署名で ds:Signature を複数持つことにより独立署名を実現することができる。

2.3. 複数署名がカウンタ署名、または独立署名とカウンタ署名が混在する場合

署名の順序が問題となる場合には、CMS 署名では CounterSignature 属性、XML 署名では CounterSignature プロパティを使用することができる。CounterSignature 属性やプロパティはこれ自体を複数持つことができ、また、属性やプロパティの値としても複数の署名を持つことができる。

例えば、署名の関係が担当者、係長、課長、部長といった一つの線で表される承認順序であった場合、CounterSignature を入れ子構造で利用することにより、これを実現することができる。

木構造となるような複雑な署名の関係があるような場合、たとえ下図に示すような署名の関係がある場合には、CMS 署名で CounterSignatures 属性の属性である署名者署名情報(CMS ならば SignerInfo)を複数持たせることにより、ある署名に対し複数の人が並列にカウンタ署名を行うことができる。

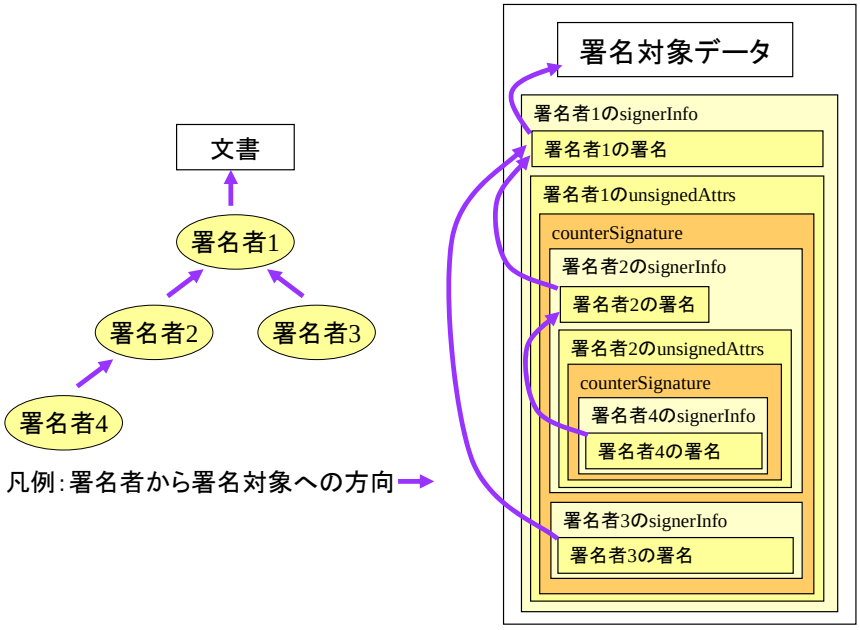


図 2.3.1: CMS 署名による木構造のカウンタ署名の例

2.4. CounterSignature 属性/プロパティがある場合の CAdES/XAdES-T

最初の署名者の署名に対して署名時刻を確定するために署名タイムスタンプを付与するのと同様にカウンタ署名を行った署名者の署名に対しても署名タイムスタンプを付与することができる。本書ではカウンタ署名の検証が重要な意味を持つ場合にはカウンタ署名に対する署名タイムスタンプを付与することを推奨する。

カウンタ署名を付与してから全ての署名タイムスタンプを付与するか、また逆に署名者の署名タイムスタンプを付与した後にカウンタ署名を付与しカウンタ署名の署名タイムスタンプを付与するかは、CAdES-T でも XAdES-T でも、どちらもサポートしている。

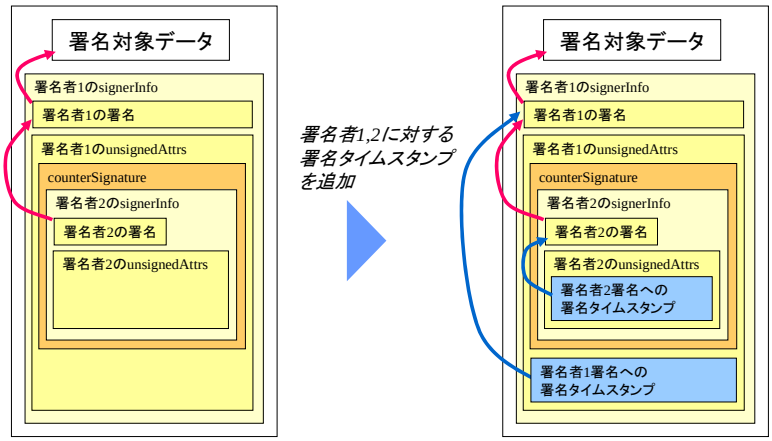


図 2.4.1: CAdES において先にカウンタ署名がある場合の署名タイムスタンプの追加

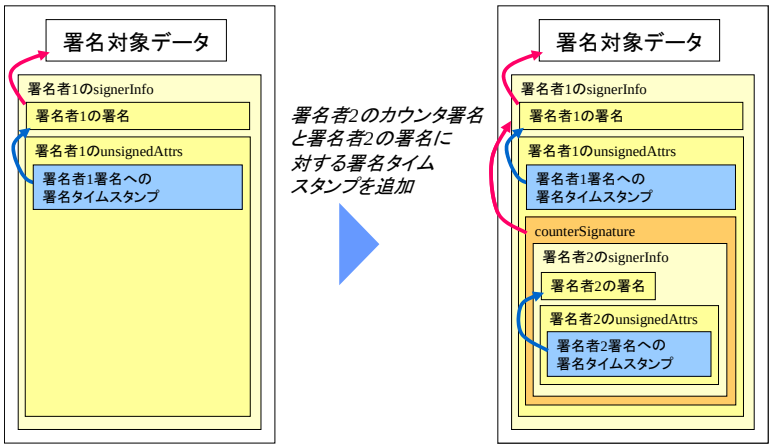


図 2.4.2: CAdES においてカウンタ署名とそのタイムスタンプを後で追加する場合

前述の 2 つの図で署名者 1 の SignerInfo の unsignedAttrs の要素の順序がカウンタ署名と署名タイムスタンプとで入れ替わることがあることに注意されたい。

2.5. CounterSignature 属性/プロパティがある場合の検証情報の格納 (CAdES/XAdES-C/X)

CounterSignature 属性/プロパティがある場合、証明書チェーンや失効情報などの検証情報の格納は通常の署名者署名情報の格納とほぼ同じであり CAdES や XAdES の検証情報の格納に従うが、CAdES の場合には、certificates フィールドや crls フィールドには CounterSignature の検証情報は格納できないため、非署名属性/プロパティに格納することとなるので、注意が必要である。

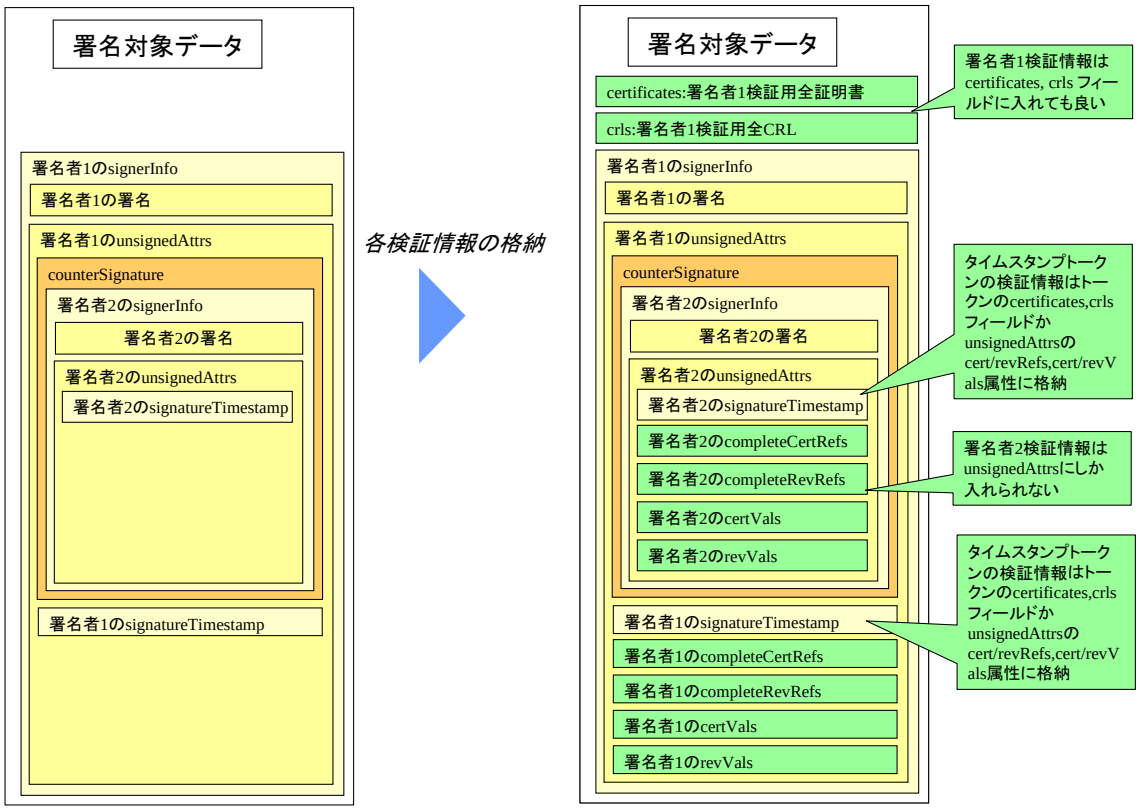


図 2.5.1: CAdES でカウンタ署名がある場合の証明書検証情報の格納

2.6. CounterSignature 属性/プロパティがある場合のアーカイブタイムスタンプ(CAdES/XAdES-A)

アーカイブタイムスタンプは、そのアーカイブタイムスタンプ以降に付与されたアーカイブタイムスタンプ属性を除く全ての非署名属性/プロパティを保護することができるため、CounterSignature 属性/プロパティに対して特別な処理を必要としない。これに従い、CounterSignatures 属性/プロパティに対してアーカイブタイムスタンプを付与する必要はなく、一つの世代全体で 1 つのアーカイブタイムスタンプを付与すればよい。アーカイブタイムスタンプを付与する前は、いかなる属性/プロパティも追加/削除/変更することができるが、アーカイブタイムスタンプを付与した後はアーカイブタイムスタンプ以外の属性/プロパティを追加してはならない。

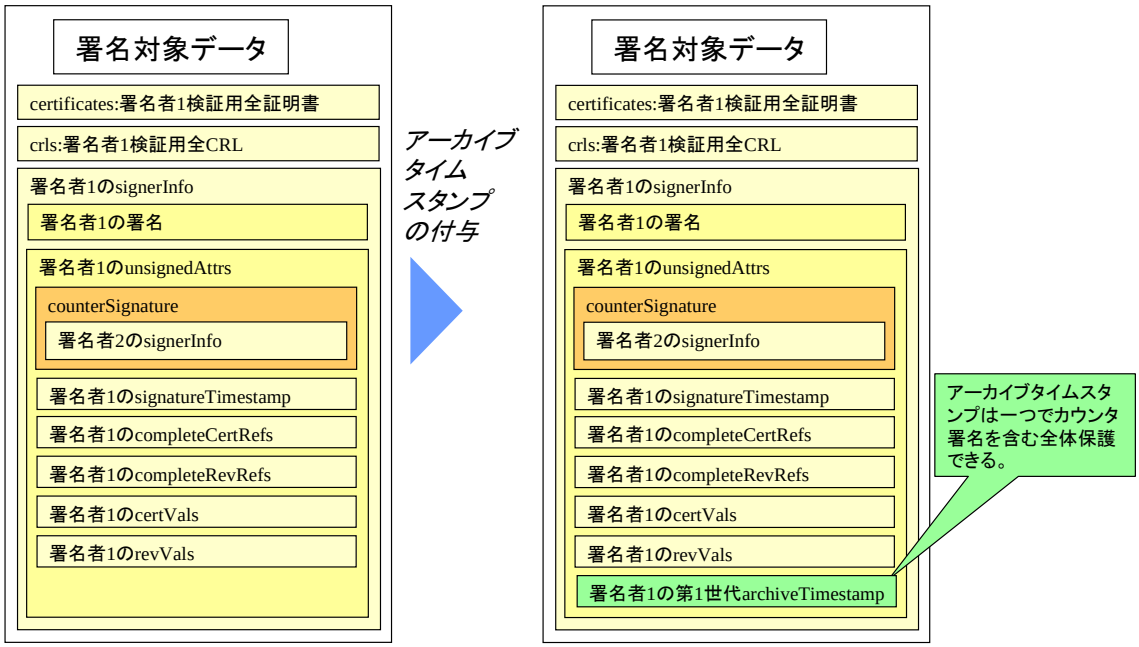


図 2.6.1: CAdES におけるカウンタ署名がある場合のアーカイブタイムスタンプの付与

.

3. 付録３：ハッシュアルゴリズムのパラメータの扱い

CMS SignedData あるいは CAdES の利用者および実装者は、ハッシュアルゴリズムを指定する場合と署名値を計算する場合とで、ハッシュアルゴリズムのパラメータの扱い方が異なることに注意する必要がある。RSA 公開鍵暗号に基づく署名アルゴリズムを使用して署名値を計算する際には、パディング処理においてハッシュアルゴリズムのパラメータを使用するためである。[27][28][12][29]

これらの相違が、相互運用性を阻害する要因となることがある。本節では、ハッシュアルゴリズムのパラメータの扱い方の差異の主要なポイントについて説明する。

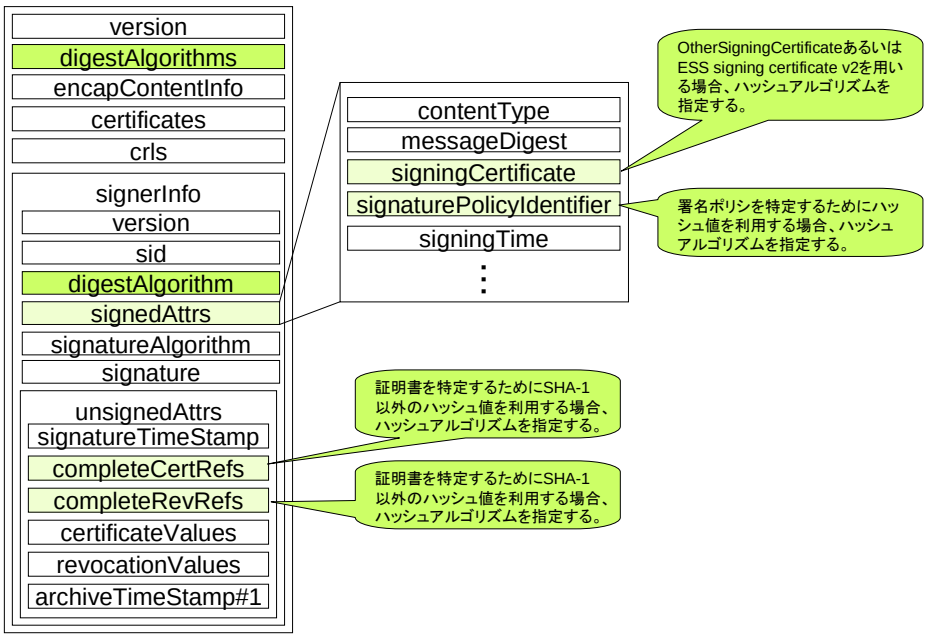
3.1 ハッシュアルゴリズムを指定する場合

ハッシュアルゴリズム（ダイジェストアルゴリズム）は AlgorithmIdentifier 型で指定する。AlgorithmIdentifier の型定義は次のとおりである。

```
RFC 3280 4.1.1.2 節 signatureAlgorithm より
AlgorithmIdentifier  ::=  SEQUENCE  {
    algorithm             OBJECT IDENTIFIER,
    parameters            ANY DEFINED BY algorithm OPTIONAL  }
```

parameters フィールドには、使用するアルゴリズムが必要とするパラメータを格納することになっている。ところが標準的なハッシュアルゴリズムはパラメータを必要としない。パラメータ不要の場合の parameters フィールドの取扱いに一貫性がないことにより、実装に混乱がもたらされる可能性がある。

CAdES（CMS SignedData を含む）においてハッシュアルゴリズム（ダイジェストアルゴリズム）を指定するフィールドは複数存在する（図にその一部を示す）。



CAdES（CMS SignedData を含む）で次表に示すハッシュアルゴリズムを指定する場合には注意が必要である。

ハッシュアルゴリズム	ハッシュアルゴリズムの OID
MD2	Md2 ::= {iso(1) member-body(2) us(840) rsadsi(113549) digestAlgorithm(2) 2}
MD5	md5 ::= {iso(1) member-body(2) us(840) rsadsi(113549) digestAlgorithm(2) 5}
SHA-1	sha-1 ::= {iso(1) identified-organization(3) oiw(14) secsig(3) algorithm(2) 26}
SHA-224	sha224 ::= {joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistalgorithm(4) hashalgs(2) 4}
SHA-256	sha256 ::= {joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistalgorithm(4) hashalgs(2) 1}
SHA-384	sha384::= {joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistalgorithm(4) hashalgs(2) 2}
SHA-512	sha51::= {joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistalgorithm(4) hashalgs(2) 3}

MD2 あるいは MD5 を指定する場合、parameters フィールドは存在しなければならず、

その値に NULL を設定しなければならない([8]2.2)。一方、SHA-1、SHA-224、SHA-256、SHA-384、SHA-512 を指定する場合、parameters フィールドを省略するのが正しい実装である([8]2.1,[28],[12]2.1)。ただし、以前より存在する多くの実装との互換性を保つため、双方の場合において、parameters フィールドが存在しないデータ、parameters フィールドが存在し、値として NULL が設定されているデータともに受け入れるよう実装しなければならない([8]2.1,[28],[12]2.1)。

3.2 RSASSA-PKCS1-v1_5 を計算する場合

RSA 公開鍵暗号に基づく署名アルゴリズムは PKCS #1[27]においてRSASSA-PKCS1-v1_5 と RSASSA-PSS の 2 つが規定されている。RSASSA-PSS は数学的に安全であることが証明された改良版であるが、現時点では RSASSA-PKCS1_v1_5 が一般的に利用されている。

次表に示す RSA 暗号に基づく署名アルゴリズムを利用する場合のハッシュアルゴリズムのパラメータの取扱いには、ハッシュアルゴリズムを指定する場合とは異なる注意を要する。

ハッシュアルゴリズム	対応する署名アルゴリズムの OID
MD2	md2WithRSAEncryption ::= {pkcs-1 2}
MD5	md5WithRSAEncryption ::= {pkcs-1 4}
SHA-1	sha1WithRSAEncryption ::= {pkcs-1 5}
SHA-224	sha224WithRSAEncryption ::= {pkcs-1 14}
SHA-256	sha256WithRSAEncryption ::= {pkcs-1 11}
SHA-384	sha384WithRSAEncryption ::= {pkcs-1 12}
SHA-512	sha512WithRSAEncryption ::= {pkcs-1 13}

{pkcs-1} ::= {iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1)}

RSASSA-PKCS1-v1_5 を計算する場合、RSA 秘密鍵演算の対象となるデータ(EM とする)は、EMSA-PKCS1-v1_5 として定義されるエンコード方式に従って計算される。

$$EM = 0x00 \,||\, 0x01 \,||\, PS \,||\, 0x00 \,||\, T\,.$$

PS は 8 つ以上の 0xff からなる 16 進数文字列、T は DigestInfo を DER エンコードした値である。

```
DigestInfo ::= SEQUENCE {
                    digestAlgorithm AlgorithmIdentifier
                    digest OCTET STRING
}
```

このときの digestAlgorithm における parameters フィールドの取扱いが問題となる。digestAlgorithm に MD2、MD5、SHA-1、SHA-256、SHA-384、SHA-512 を利用する場合の parameters フィールドには NULL 必ず設定する必要があり、このフィールドを省略してはならない。検証処理においても、parameters フィールドが存在しない場合にはエラーとすべきである。

参考までに各ハッシュアルゴリズムを利用したときの T の値を次に示す。ただし、H は各ハッシュアルゴリズムを対象メッセージに対して適用したときのハッシュ値である。

MD2: **(0x)30 20 30 0c 06 08 2a 86 48 86 f7 0d 02 02 05 00 04 10** ‖ *H.*

MD5: **(0x)30 20 30 0c 06 08 2a 86 48 86 f7 0d 02 05 05 00 04 10** ‖ *H.*

SHA-1: **(0x)30 21 30 09 06 05 2b 0e 03 02 1a 05 00 04 14** ‖ *H.*

SHA-224:**(0x)30 2d 30 0d 06 09 60 86 48 01 65 03 04 02 04 05 00 04 1c** ‖ *H.*

SHA-256:**(0x)30 31 30 0d 06 09 60 86 48 01 65 03 04 02 01 05 00 04 20** ‖ *H.*

SHA-384:**(0x)30 41 30 0d 06 09 60 86 48 01 65 03 04 02 02 05 00 04 30** ‖ *H.*

SHA-512:**(0x)30 51 30 0d 06 09 60 86 48 01 65 03 04 02 03 05 00 04 40** ‖ *H.*

アンダーラインを付した “05 00” が parameters に NULL が設定されていることを示す。

3.3 まとめ

付録３の内容を次表にまとめる。

	ハッシュアルゴリズム	生成		検証	
		parameters フィールド		parameters フィールド	
		なし	NULL 設定	なし	NULL 設定
アルゴリズム指定	MD2, MD5	×	○	○	○
	SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	○	×	○	○
RSASSA-PKCS1-v1_5 計算	MD2, MD5, SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	×	○	×	○

×：MUST NOT　○：MUST

参考文献

[1] RFC 2560 X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP, Jun 1999, M.Myers, et al., http://www.ietf.org/rfc/rfc2560.txt

[2] RFC 2634 Enhanced Security Services for S/MIME, Jun 1999, P. Hoffman, http://www.ietf.org/rfc/rfc2634.txt

[3] RFC 3125 Electronic Signature Policies, Sep 2001, J. Ross et. al, http://www.ietf.org/rfc/rfc3125.txt

[4] RFC 3126 Electronic Signature Formats for long term electronic signatures, Sep 2001, D.Pinkas et. al, http://www.ietf.org/rfc/rfc3126.txt

[5] RFC 3161 Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP), Aug 2001, C.Adams et al, http://www.ietf.org/rfc/rfc3126.txt

[6] RFC 3280 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, Apr 2002, R.Housley, et al., http://www.ietf.org/rfc/rfc3280.txt

[7] RFC 3369 Cryptographic Message Syntax (CMS), Aug 2002, R.Housley, http://www.ietf.org/rfc/rfc3369.txt

[8] RFC 3370 Cryptographic Message Syntax (CMS) Algorithms, Aug 2002, R.Housley, http://www.ietf.org/rfc/rfc3370.txt

[9] RFC 3560 Use of the RSAES-OAEP Key Transport Algorithm in the Cryptographic Message Syntax (CMS), Jul 2003, R. Housley, http://www.ietf.org/rfc/rfc3560.txt

[10] RFC 3852 Cryptographic Message Syntax (CMS), Jul 2004, R.Housley, http://www.ietf.org/rfc/rfc3852.txt

[11] RFC 4051 Additional XML Security Uniform Resource Identifiers (URIs), Apr 2005, D.Eastlake 3rd, http://www.ietf.org/rfc/rfc4051.txt

[12] RFC4055 Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, Jun 2005, J. Schaad, et al., http://www.ietf.org/rfc/rfc4055.txt

[13] Internet Draft CMS Advanced Electronic Signatures (CAdES) - Expired, J.Ross, et al., Dec 2005,
http://www.ietf.org/internet-drafts/draft-ietf-smime-cades-01.txt

[14] Internet Draft ESS Update: Adding CertID Algorithm Agility, J.Schaad, Jan 2007, http://www.ietf.org/html.charters/smime-charter.html

[15] ETSI TS 101 733 V1.4.0 (2002-09) Electronic Signatures and Infrastructures (ESI); Electronic Signature Formats, Sep 2002, ETSI

[16] ETSI TS 101 733 V1.5.1 (2003-12) Electronic Signatures and Infrastructures (ESI);

Electronic Signature Formats, Sep 2002, ETSI
[17] ETSI TS 101 733 V1.6.3 (2005-09) Electronic Signatures and Infrastructures (ESI); CMS Advanced Electronic Signatures (CAdES), Sep 2005, ETSI
[18] ETSI TS 101 733 V1.7.3 (2007-01) Electronic Signatures and Infrastructures (ESI); CMS Advanced Electronic Signatures (CAdES), Jan 2007, ETSI
[19]ETSI TS 101 903 V1.1.1 (2002-02) XML Advanced Electronic Signatures (XAdES), Feb 2002, ETSI
[20] ETSI TS 101 903 V1.2.2 (2004-04) XML Advanced Electronic Signatures (XAdES), Apr 2004, ETSI
[21] ETSI TS 101 903 V1.3.1 (DRAFT) XML Advanced Electronic Signatures (XAdES), DRAFT, ETSI
[22] ETSI TS 101 903 V1.3.2 (2006-03) XML Advanced Electronic Signatures (XAdES), Mar 2006, ETSI
[23] ETSI XAdES PLUGTESTS Final Report Draft, 2003, M.Centner, et.al., http://www.etsi.org/plugtests/History/DOC/XAdESFinalReport.pdf
[24] ITU-T X.509 Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks, Aug 2005, ITU-T
[25] ITU-T X.680 Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation, Jul 2002, ITU-T
[26] ITU-T X.690 Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER), Jul 2002, ITU-T
[27] PKCS #1 v2.1: RSA Cryptography Standard, Jun 2002, RSA Laboratories, ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf
[28] PKCS #1 v2.1 Errata Revision 2.0, Dec 2005, RSA Laboratories, ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1errata.txt
[29] PKCS #1: RSA Cryptography Standard, RSA Laboratories, http://www.rsa.com/rsalabs/node.asp?id=2125
[30] PKCS #7: Cryptographic Message Syntax Standard, An RSA Laboratories Technical Note, Version 1.5, Nov 1993, RSA Laboratories
[31] XML Advanced Electronic Signatures (XAdES), W3C Note 20 Feb 2003, http://www.w3.org/TR/XAdES/
[32] XML-Signature Syntax and Processing, W3C Recommendation 12 Feb 2002, http://www.w3.org/TR/xmldsig-core/
[33] Canonical XML Version 1.0, W3C Recommendation 15 Mar 2001, http://www.w3.org/TR/xml-c14n

[34] Exclusive XML Canonicalization Version 1.0, W3C Recommendation 18 July 2002, http://www.w3.org/TR/xml-exc-c14n/
[35] XML Path Language (XPath) Version 1.0, W3C Recommendation 16 November 1999, http://www.w3.org/TR/xpath
[36] XML Pointer Language (XPointer), W3C Working Draft 16 August 2002, http://www.w3.org/TR/xptr
[37] XSL Transformations (XSLT) Version 1.0, W3C Recommendation 16 November 1999, http://www.w3.org/TR/xslt
[38] XML Encryption Syntax and Processing, W3C Recommendation 10 December 2002, http://www.w3.org/TR/xmlenc-core/
[39] 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), 2001, http://www.meti.go.jp/policy/netsecurity/digitalsign.htm
[40] 民間事業者等が行う書面の保存等における情報通信の技術の利用に関する法律(平成 16 年法律第 149 号), 2005, http://www.kantei.go.jp/jp/singi/it2/hourei/16-149gou/honbun.html
[41] e-文書法におけるタイムスタンプ適用ガイドライン,平成 17 年 10 月,タイムビジネス推進協議会, http://www.scat.or.jp/time/PDF/tekiyouguidelineVer1.1.pdf
[42]平成 16 年度 EC 技術基盤の相互運用性に関する調査研究(電子署名生成・検証システムのセキュリティ環境の国際標準化等の調査)電子文書の長期保存と見読性に関するガイドライン,平成 17 年 2 月,電子商取引推進協議会・財団法人日本情報処理開発協会電子商取引推進センター
[43]平成 15 年度 EC 技術基盤の相互運用性に関する調査研究(電子署名生成・検証システムのセキュリティ環境の国際標準化等の調査)署名ポリシー調査報告書,平成 16 年 3 月,電子商取引推進協議会・財団法人日本情報処理開発協会電子商取引推進センター
[44]平成 15 年度 EC 技術基盤の相互運用性に関する調査研究(電子署名生成・検証システムのセキュリティ環境の国際標準化等の調査)電子署名文書長期保存に関する実用化動向調査報告書,平成 16 年 3 月,電子商取引推進協議会・財団法人日本情報処理開発協会電子商取引推進センター
[45]平成 15 年度 EC 技術基盤の相互運用性に関する調査研究(電子署名生成・検証システムのセキュリティ環境の国際標準化等の調査)電子文書の長期保存と見読性に関する調査報告書,平成 16 年 3 月,電子商取引推進協議会・財団法人日本情報処理開発協会電子商取引推進センター
[46]平成 14 年度 EC 技術基盤の相互運用性に関する調査研究(電子署名生成・検証システムのセキュリティ環境の国際標準化等の調査)タイムスタンプサービス調査報告書,平成 15 年 3 月,電子商取引推進協議会・財団法人日本情報処理開発協会電子商取引推進センター
[47]平成 14 年度 EC 技術基盤の相互運用性に関する調査研究(電子署名生成・検証システム

のセキュリティ環境の国際標準化等の調査)タイムスタンプサービス利用ガイドライン,平成 15 年 3 月,電子商取引推進協議会・財団法人日本情報処理開発協会電子商取引推進センター
[48]平成 14 年度 EC 技術基盤の相互運用性に関する調査研究(電子署名生成・検証システムのセキュリティ環境の国際標準化等の調査)タイムスタンプサービス運用ガイドライン,平成 15 年 3 月,電子商取引推進協議会・財団法人日本情報処理開発協会電子商取引推進センター
[49]電子署名文書長期保存に関するガイドライン,平成 14 年 3 月,電子商取引推進協議会 認証・公証ワーキンググループ
[50]電子署名文書長期保存に関する中間報告,平成 13 年 3 月,電子商取引推進協議会 認証・公証ワーキンググループ
[51]CMS 長期署名プロファイル, 2005 年 8 月, 次世代電子商取引推進協議会(ECOM), http://www.ecom.jp/report/electronic_signatures/CMSformat.pdf
[52]XAdES 長期署名プロファイル, 2005 年 8 月, 次世代電子商取引推進協議会(ECOM), http://www.ecom.jp/report/electronic_signatures/XAdESLong-TermSignatureFormatProfile_V0.6pub_.pdf
[53] 暗号メッセージ構文を利用した電子署名(CAdES)の長期署名プロファイルに関する要求事項 JIS 原案, 2006 年 12 月, 次世代電子商取引推進協議会(ECOM), http://www.ecom.jp/report/JIS_CAdES_Profile.pdf
[54] 拡張可能なマーク付け言語を利用した電子署名(XAdES)の長期署名プロファイルに関する要求事項 JIS 原案, 2006 年 12 月, 次世代電子商取引推進協議会(ECOM), http://www.ecom.jp/report/JIS_XAdES_Profile.pdf

第二部

電子文書長期保存

まえがき

企業、官公庁、自治体を取り巻く環境において、「情報公開法」、「PL 法」、「個人情報保護法」、「ISO 9001」、「ISO 14000」などに加え、「金融商品取引法」（いわゆる、JSOX 法）の 2006 年 8 月制定、2006 年 11 月の「財務報告に係わる内部統制の評価及び監査の実施基準－公開草案－」[1]提示により、記録、保管、検索、参照・検証などの「記録管理」の必要性が増している。企業、官公庁・自治体が作成・生成する電子文書・情報をその組織の内部・外部の利用のために保管し、利用していくためには、以下の 3 項目に留意し、「記録管理システム」を整備していく必要がある。

（1）　法律、規制、適用標準、組織方針などの"記録管理システム"への「外部からの要請」
（2）　人的管理も含めた「記録・保存マネジメントシステム」
（3）　文書生成、保管などに係わる"記録管理システム"の「技術水準・動向」

本報告では、これらの事項について解説、説明を行う。

"記録管理システム"への「外部から要請事項」の例として内部統制および JSOX 法を取り上げ、その要請事項について説明、解説等を行う。次に、「記録・保存マネジメントシステム」としての ISO 15489 について、解説を行う。短期の記録・保存における「長期署名フォーマット」の適用意義について説明し、最後に「電子保管媒体の動向」について説明する。

1. JSOX 法と記録管理

米国のエンロン事件に端を発し、その後、日米で多発した企業の粉飾決算、不正会計等が各種の規制強化を生み出し、内部統制導入の流れを生み出している。

米国では、企業改革法（サーベンス・オクスレイ法＝SOX 法）が 2002 年 7 月に制定された。日本では、会社法が 2006 年 5 月施行され、金融商品取引法(JSOX 法)は 2006 年 6 月に制定され、2008 年度決算から義務付けられることとなった。しかしながら、新会社法、JSOX 法ともに、その内部統制の証として記録の保存を必要としているが、会社規模、業種の事情などを配慮し、最低限の要請を定めているに過ぎない。これらの法律が、「企業価値の維持・向上」を目指していることからすると各企業経営者に、求められているのは法律を最低限クリアすることだけではない。本来の目的である、「企業価値の維持・向上」のために各社がどこまでの対応をしていくかは、監査法人ではなく、各企業の経営者に託されていることに留意べきと考える。様々な不祥事に見舞われている企業でさえも ISO 9001、ISO 14000 を取得していることから経営者の責任を監査人や認証機関に委ねるべきでないことは自明である。本章では JSOX と記録管理について解説をする。

1.1. JSOX 法に見る法的要件

JSOX の実施基準「財務報告に係る内部統制の評価及び監査に関する実施基準」の公開草案[1]が 2006 年 11 月 21 日金融庁 企業会計審議会内部統制部会から公開され、パブリックコメントが募集された。2007 年 2 月 15 日に、企業会計審議会内部統制部会はパブリックコメントを踏まえ、実施基準[8]を公開した。

実施基準(公開草案)は財務報告に係る内部統制のガイドラインとされ、その構成は、

□. 内部統制の基本的枠組み

□. 財務報告に係る内部統制の評価及び報告

□. 財務報告に係る内部統制の監査

の 3 章から構成されている。「□.の内部統制の基本的枠組」の中で、内部統制の定義(目的)について以下のように定めている。

“内部統制の定義(目的)とは、基本的に業務の有効性及び効率性、財務報告の信頼性、事業活動に関わる法令等の遵守並びに資産の保全の 4 つの目的が達成されていることの合理的な保証を得るために、業務に組み込まれ、組織内のすべてのものによって遂行されるプロセスをいい、統制環境、リスクの評価と対応、統制活動、情報と伝達、モニタリング(監視活動)及び IT(情報技術)への対応の 6 つの基本的要素から構成される。”

その枠組みは、米国の COSO フレームワークをベースに日本独自の要素である資産の保全とＩＴ(情報技術)への対応を加えている。この 4 つの目的と 6 つの基本的な要素という内部統制の枠組みを成立するためには手続きに従った統制確保とレビューのための文書化が必須条件となってくる。公開草案では記録管理の基準については「□. 財務報告に係わる内

部統制の評価及び報告」で、記述している。以下に記録保存対象文書と記録保存要件について抜粋する。

1.-1　記録保存すべき文書の例

イ．財務報告に係わる内部統制の整備及び運用の方針及び手続き

ロ．全社的な内部統制の評価にあたって、経営者が採用する評価項目ごとの整備及び運用の状況

ハ．重要な勘定科目や開示項目に関連する業務プロセスの概要（各業務プロセスにおけるシステムに関する流れやＩＴに関する業務処理統制の概要、使用しているシステム一覧）

ニ．各業務プロセスにおいて重要な虚偽表示が発生するリスクとそれを低減する内部統制の内容

（実在性、網羅性、権利と義務の帰属、評価の妥当性、期間配分の適切性、表示の妥当性との関連を含む。また、ＩＴを利用した内部統制の内容を含む。）

ホ.上記ニ.に係わる内部統制整備及び運用の状況

ヘ.財務報告に係わる内部統制の有効性の評価手続並びに発見した不備及びその是正措置

・評価計画に関する記録

・評価範囲の決定に関する記録（評価の範囲に関する決定方法及び根拠等を含む）

・実施した内部統制の評価の手順及び評価結果、是正措置等に係わる記録

1-2　記録保存要件

（1）保存期間

有価商品取引法上は、有価証券報告書及びその添付書類の縦覧期間（５年）と同等

（2）保存方法

磁気媒体、紙又はフィルム等により保存

後日、第３者による検証が可能となるよう、関連する証拠書類もあわせて保存。

また。2007 年 1 月 19 日、経済産業省は JSOX を念頭においたＩＴ統制に関する、概念、経営者評価、導入ガイダンスとして「システム管理基準 追補版（財務報告に係わるＩＴ統制ガイダンス）」（案）[2]を示しており、その中で記録の保存について以下のように記している。

電子的な記録は、評価後にシステム上の設定誤りにより上書き又は削除されてしまう可能性もある。また、意図的な改ざん等を考慮して適切な記録の保存が望まれる。評価結果や関連する証拠書類等の記録を適切に保存する。例えば、関連する証拠書類の紙への印刷や上書きできない媒体への保存などが考えられる。また、改ざん防止のために、電子署名等を利用する方法も考えられる。

経営者評価の結果は、後日、監査人による監査が可能となるように適切に記録しておくこと。尚、どのような評価結果を残しておくべきか、評価時に利用した関連書類はどの範囲まで残しておくべきか、それらをどのように保管しておくか（例えば、電子的記録の場合、印刷して保存するか、改ざん防止のための対策をするか）等について、事前に監査法人と協議を行っておくこと。

1.2. JSOX 法実施基準公開草案、ガイダンスに対する見解

本節では本 WG 内で検討された公開草案に対する見解を示す。記録・保存対象文書としては、内部統制整備・運用・評価のための書類、証憑など全てを対象としている。後日、第３者による検証が可能となるように、関連する証拠書類も保存することを求められていることから、統制に使用する証憑などは評価のために抽出したサンプルだけを残すのではなく、作成、発生した証憑すべてを残すことが義務付けられている。

保存期間は作成・発生時からの起算ではなく、財務報告時点から 5 年以上であり、作成・発生時点から起算すると保存期間は、通常６年以上程度が必要になってくる。保存媒体としては、「磁気媒体、紙又はフィルム等」として特に、こだわらないことを示しているが、5～6 年程度の期間であれば、敢えてフィルム(マイクロフィルム)とするまでのコストをかける必要性もなく、磁気媒体を含む各種電子媒体または紙を使用することで、対応できると考える。むしろ、適切に保存、後日、第 3 者による検証が可能であることを求めていることに注意する必要がある。これは可読性を要求していると解釈できる。

公開草案、ガイダンス双方において、具体的な指標が明記されていないことから経営者自らの判断で、記録・保存に関する規範を設け、それを監査法人の合意を取っていく必要がある。本法律の趣旨からすれば、合理的な範囲での対応で可とはなっているが、経営者が結果責任を取る必要があり、後日、容易に改ざんや廃棄などが行われたとなると経営者の責任が問われかねないことになる。したがって、実施基準に具体的な手法が明示されていないからといって、記録・保存を軽視することは避けるべきである。

1.3. JSOX 法実施基準公開草案パブリックコメント

ＪＳＯＸ実施基準公開草案に対し、ECOM から提出したパブリックコメントを以下に示す。これらの指摘に対し、2007 年 2 月 15 日の実施基準を見る限り、明確な回答、修正はなされていない。しかしながら、指摘箇所に関し、実施基準で修正記述のあった事項につ

いては、その修正内容を紹介する。

1.3.1. パブリックコメント１

（１）指摘箇所

□.内部統制の基本的枠組み　シート 17

”イ.統制環境の有効性を確保するためのＩＴの利用　”において、以下の記述があります。

「経営者や組織の構成員等が電子メール等を用いることにより、容易に不正を共謀すること等も可能としかねず、これを防止すべく組織内の通信記録の保全など適切な統制活動が必要になることにも留意する必要がある。」

（２）指摘内容

本記述には以下の２点の不明瞭な点があると考えますので、明確化をお願い致します。

□本記述は、メールを保管することを求めているか、否か。

□”通信記録の保全など”とは、メール以外の電話の通話記録、メモ用紙も含むか、否か。

（３）実施基準修正記述

「経営者や組織の構成員等が電子メール等を用いることにより、容易に不正を共謀すること等も可能としかねず、これを防止すべく適切な統制活動が必要になることにも留意する必要がある。」

1.3.2. パブリックコメント２

（１）指摘箇所

□.内部統制の基本的枠組み　シート 18

“ハ.　統制活動の有効性を確保するための IT の利用”において、以下の記述があります。

「プログラムの不正な改ざんや不正な使用等・・・　、適切なアクセス管理等の措置を講じておくことにつき留意する必要がある。」

（２）指摘内容

統制の有効性を担保するには、アクセス管理に加え、記録の改ざん防止、削除の防止、改ざんの検知も必要であります。　現記述ではアクセス管理にばかり気を取られることになりかねないため、記録の改ざん防止、削除防止、改ざん検知についても言及すべきであると考えます。"

（３）実施基準修正記述

修正無し

1.3.3. パブリックコメント３

（１）指摘箇所

Ⅱ.財務報告に係わる内部統制の評価及び報告　シート 27

②　”記録の保存”において、以下の記述があります。

「金融商品取引法上は、有価証券報告書及びその添付書類の縦覧期間（５年）を勘案して、それと同程度の期間、適切な範囲及び方法(磁気媒体、紙またはフィルム等）により保存することが考えられる。」

（２）指摘内容

保管期限５年”は、敢えてフィルム（マイクロフィルム）を使用しなければいけない程の長期の保管ではなく，例示として適切ではないと考えます。また、磁気媒体を例にあげるよりも光ディスク、磁気ディスクアレイ装置なども含めて、電子記録媒体／電子記録装置と例示する方が望ましいと考えます。

（３）実施基準修正記述

「金融商品取引法上は、有価証券報告書及びその添付書類の縦覧期間（５年）を勘案して、それと同程度の期間、適切な範囲及び方法(磁気媒体、紙またはフィルム等のほか必要に応じて適時に可視化する方法）により保存することが考えられる。」

【挿入：ほか必要に応じて適時に可視化する方法】

1.3.4. パブリックコメント４

（１） 指摘箇所

Ⅱ.財務報告に係わる内部統制の評価及び報告　シート 21

a.IT に係わる全般統制の評価”においては、以下の記述があります。

・内外からのアクセス管理などのシステム安全性の確保

（２）指摘内容

システムの安全性の確保には、アクセス管理に加え、WORM(Write Once Read Many)機構による上書きや削除の防止なども有効であり、記録の改ざん防止、削除防止についても言及すべきであると考えます。"

（３）実施基準修正記述

修正無し

1.3.5. パブリックコメント５

（１） 指摘箇所

Ⅰ.内部統制の基本的枠組み　シート 19

イ.組織目標を達成するための IT の統制目標”において、以下の記述があります。

「ｃ． 信頼性：情報が組織の意思・意図に沿って承認され、漏れなく正確に記録・処理されること（正当性・完全性・正確性)」

（２） 指摘内容

信頼性の確保のためには、これに加え、適切な記録の保管が必要であると考えます。

この観点を追加し、以下のような表現にすることを提案致します。

「ｃ． 信頼性：情報が組織の意思・意図に沿って承認され、漏れなく正確に記録・処理さ

れ、改ざん・削除・欠損などを適切に防止しながら保管されること。」

（３）　実施基準修正記述

ａ）該当箇所：修正なし。

ｂ）Ⅱ.財務報告に係わる内部統制の評価及び報告　シート 27

②　”記録の保存

「記録･保存に当たっては、後日、第三者による検証が可能となるよう、関連する証拠書類を適切に保存する必要がある。」【変更：あわせて→適切に】

1.4.　JSOX 法システム管理基準追補版パブリックコメント

「システム管理基準 追補版（財務報告に係わるＩＴ統制ガイダンス)」（経済産業省）[2]に対し、ECOM から提出したパブリックコメントを以下に示す。

1.4.1. パブリックコメント 1

（1）指摘箇所

2.　記録保存

「電子的な記録は、評価後にシステム上の設定誤りにより上書き又は削除されてしまう可能性もある。また、意図的な改ざん等を配慮して適切な記録の保存が望まれる。評価結果や関連する証拠書類等の記録を適切に存する。例えば、関連する証拠書類の紙への印刷や、上書きできない保存媒体への保存なども考えられる。」

（１）　指摘内容 1

「・・・評価結果や関連する証拠書類等の記録を適切に存する・・・」における「存する」は「保存する」の誤記である。

（２）　指摘内容 2

「・・・関連する証拠書類の紙への印刷・・・」における印刷の必要性が曖昧である。改ざん防止という意味では、5 年程度の保管ではむしろ紙の方が差し替えなどのリスクが高い。これを防ぐには物理的なアクセス制限手段（保管庫、アクセス記録など）を準備せねばならず、費用的にも、運用的にもあまり有効な手段とは言えない。むしろ、一般的な文書管理におけるアクセス権の管理、アクセス履歴保存の方がセキュリティが高いと考える。

1.5.　JSOX 法記録・保存への電子保管適用の利点

JSOX 実施基準案（金融庁）[8]、「システム管理基準 追補版（財務報告に係わるＩＴ統制ガイダンス)」（経済産業省）[2]のいずれにおいても、電子保管は必須とはされていないが、逆に、電子保管自体は許されている。特に、具体的な指針がなく、方法については監査法人との相談事項になっている。

保管期限 5 年という期間という特性からすると、紙やそれを変換したマイクロフィルムを積極的に利用する必要性はなく、コスト低減が図れる場面では、むしろ、電子保管を利

用していくことが有効である。JSOX に係わる保管のコストとしては以下の通常の文書保管コスト

- 媒体コスト、保管場所コスト、検索コスト、システム維持コスト

の他に、内部統制の観点からは“保管文書登録コスト”、“運用テストコスト”の検討が望まれる。

紙をスキャナーで取り込んだ、PDF ファイルの証憑も他の法律からの縛りがなければ、JSOX では証憑になり得る。電子署名、タイムスタンプ、アクセスログなどにより、人手処理、紙処理に比べ格段に真正性が高まる。JSOX では証憑の網羅性を要求しており、人手での証憑の台帳登録、紙ファイル保管にくらべ、電子的な証憑の登録は漏れがなく網羅性が高くなる。また、運用テストにおいて、無作為なサンプル抽出の証明もやさしい、証憑の収集コストも低くなるという利点がある。実施基準の最低ラインのクリア、監査法人の合意はもちろんではあるが、各社がより信頼性の高いＩＴを利用した文書保管システムに取り組むことを期待したい。

2. 記録・保存マネジメント手法

「金融商品取引法」（JSOX 法）は、財務報告に的を絞った内部統制/ディスクロジャーを要求している。一方、「新会社法」は、コーポレートガバナンスと財務報告を含めた全般的なリスク管理体制を求めている。いずれもその枠組みは、米国の COSO フレームワークに倣った日本版のフレームワークである 4 つの目的と 6 つの基本的な要素から成り立っている。経営者の責任として、基本方針を決定し（P: Plan）、構築・運用し（D; Do）、経営者自身、監査委員会および外部監査人が評価し（C: Check）、それを開示する（A: Action）という PDCA というプロセスを廻すこと、即ち内部統制のマネジメントシステムの構築が要求されているのである。これは、ISO 9001、ISO 14000 などにおいてもそれぞれの内部統制を必要とされており、同様である。

内部統制のマネジメントシステムを運用するためには、統制のための手続きを文書化し、統制の対象となる各種議事録など文書を記録し、保存することにより内部、外部の評価に耐えられる記録管理システムの構築が求められてくる。内部統制のための記録管理の要件を ISO 15489 Records Management 記録管理に準拠して考えてみる。

2.1. 記録管理の国際標準 ISO 15489 とは

2001 年に発行された ISO 15489 Information and documentation-Records Management：記録管理は、Part1: General 総説[6]と Part2：Guideline ガイドライン[7]の 2 部から構成されている。ISO 15489-1 は、2005 年（平成 17 年）7 月 20 日に、日本の JIS X 0902-1「情報及びドキュメンテーション記録管理 第 1 部：総説」として発行されている。ISO/TR 15489-2 記録管理：ガイドラインは、2001 年に「ISO 15489-1 記録管理: 総説」と同時に発行された。その位置づけは、記録管理の専門家および組織での記録管理の責任者のための ISO 15489-1 の実施手引書にあたり、記録管理を必要とするすべての組織において ISO 15489-1 を実践するための方法を提供している。日本においては、本来、ISO 15489-1 と同時に JIS 化されることが望ましかったが 2007 年 1 月現在まだ JIS として発行されていない。現在 JIS/TR として発行準備中である。

ISO15489-1 の概要は、平成 17 年 2 月発行の ECOM 報告書「電子文書の長期保存と見読性に関するガイドライン」[3]の 1.2 節、標準化動向の中で紹介している。主な特徴には以下があげられる。

・記録管理のベストプラクティス
・官民を問わず、すべての組織の記録管理に指針を提供
・説明責任のため記録管理
・記録の品質を重視
・規格の中の規格（他の規格にも関連）
・すべての媒体、フォーマットをカバー

日本では、文書と記録の定義が曖昧であるが、ISO 15489-1 では、「文書」と「記録」を、次のように定義している。

・「文書」(document)：1 つの単位として取扱われる記録化された情報、又はオブジェクト
・「記録」(records)：法的な義務の履行、あるいは業務処理における証拠及び情報として組織又は個人が作成、取得及び維持する情報

「記録」は、一般的な文書より重要度が高く証拠性があるとされており、「文書」は修正可能だが「記録」となると修正変更が出来ない。

そして、「記録管理」については、以下のように定義している。

・　「記録管理」(records management)：記録の作成、取得、維持、利用及び処分の効率的で統制に責任を持つ管理の分野であって、記録の形で業務活動及び処理に関する証拠及び情報を取り込み、維持するためのプロセスを含む。

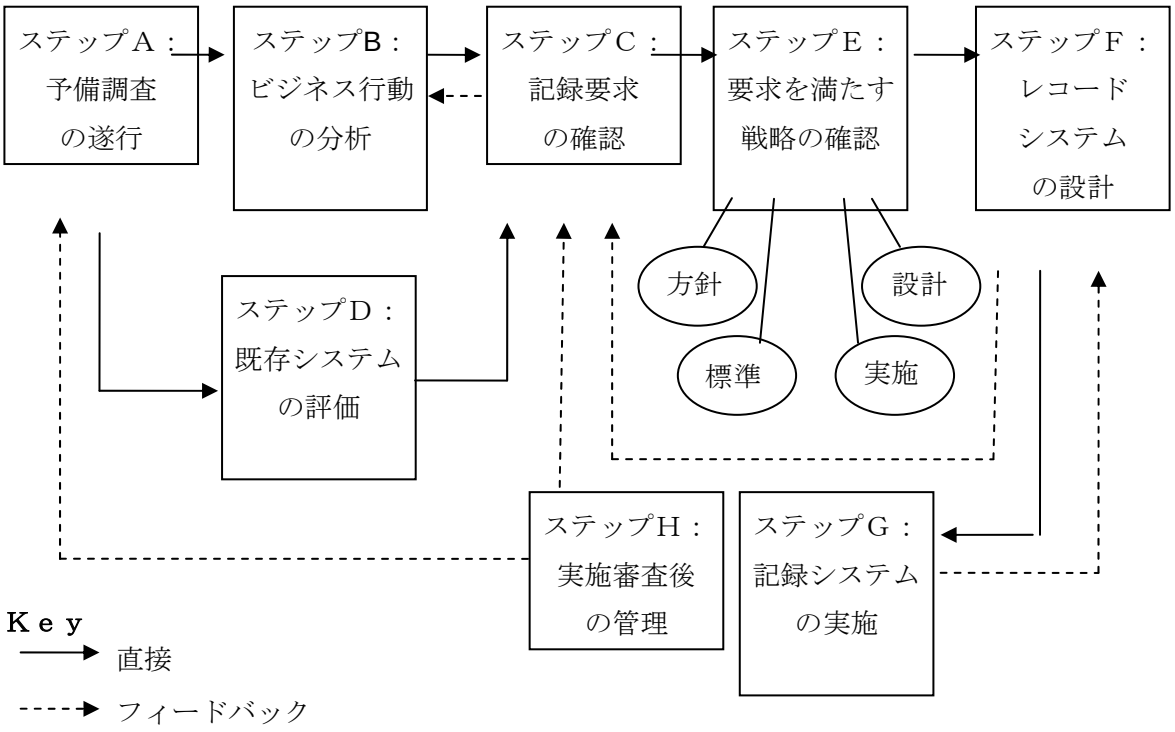
2.2. ISO 15489-2 Guidelines の紹介

ISO 15489-2 Guidelines は下記の 6 章から構成される。

１．適用範囲
２.方針と責任
３.戦略及び、設計及び実施
４.記録のプロセス及び制御
５.監視及び監査
６．研修

ISO 15489-2 Guidelines Technical Report　3 章戦略、設計及び実施においては、ステップ A から H にいたる各ステップでそれぞれの目的や成果物等について具体的に示しており、良い記録システム作りに役立つ。

ステップ A:予備調査の遂行
ステップ B:ビジネス活動の分析
ステップ C:記録要求の確認
ステップ D:既存システムの評価
ステップ E:要求を満たすための戦略の確認
ステップ F:レコードシステムの設計
ステップ G:記録システムの実施
ステップ H:実施審査後の管理



出典：オーストラリア国立公文書館及びニューサウスウエールズ州

図 2.2.1: レコードシステムの設計と実施

例えば、ステップ：B ビジネス行動の分析の目的は、組織の業務活動の概念モデルを作ることであるとし、記録が組織の業務及び業務プロセスの両者にどのように関連するかを示すものとしている。さらに、このステップ以降の記録のライフサイクルに関する意思決定や記録のアクセスに関する意思決定に役立つとしている。また、このステップの成果物として、

- 組織の業務及び業務プロセスを記述した文書
- 記録の保有期間や処分行動を規定する処分基準
- 業務分類体系　等

組織の業務と記録の関係を明らかにすることが出来るとしている。

2.3. ISO 15489 と JSOX 法における記録・保存

JSOX 法の実施基準案、ガイダンスいずれにも記録・保管のマネジメントに関する規定がない。一方、ISO 15489 は総説とガイドラインで内部統制に関する役立つ考え方や方法を示している。すなわち、個々の課題における適正化を図るだけではなく、組織全体として、業務を分析、ISO 15489-2 に示すようなステップに従い、記録・保存マネジメントを構築していくことが望ましい。

ISO15489-1 の 7.2 節、記録の特性では、記録の品質に関して次のように要求している。

『記録は、何が伝えられ、決定され、又はどのような行動がとられたかを正確に反映するとよい。記録は、それが関係する業務のニーズを支援することができ、説明責任の目的で使用することが望ましい。記録は，内容と同様に，業務処理を文書化するのに必要なメタデータを内部に含むか、それと継続的に結び付けられているか、又は関係付けられていることが望ましい。』（7.2.1　総論）

さらに、望ましい特性(良い記録の品質)として 以下の４点を、挙げており、

・「真正性」(authenticity)：権限を持った記録作成者が作成し、許可の無い記録の追加、削除、変更、利用及び隠蔽から守られること。
・「信頼性」(reliability)：信頼のおける記録とは組織の活動や業務を正確に表し、証拠が担保できる。
・「完全性」(integrity)：記録の完全性は、その内容が完結していて変更されていない。
・「利用性」(usability)：所在場所がわかり、検索でき、表示でき、解釈できる。

個々の記録・保管の手法が合致しているかを確認することも有効である。

3. 電子文書保管への長期署名フォーマットの推奨

電子文書のライフサイクルモデルについては、ECOM 報告書[3]において既に解説されている。[3]では、電子文書保管のライフサイクルモデルを図 3.1 のように示している。処理フェーズの最後に文書は確定され、管理台帳に登録し、保管媒体に記録する。保管フェーズ以降では参照のみが行われ内容の追記・変更などは行われない。

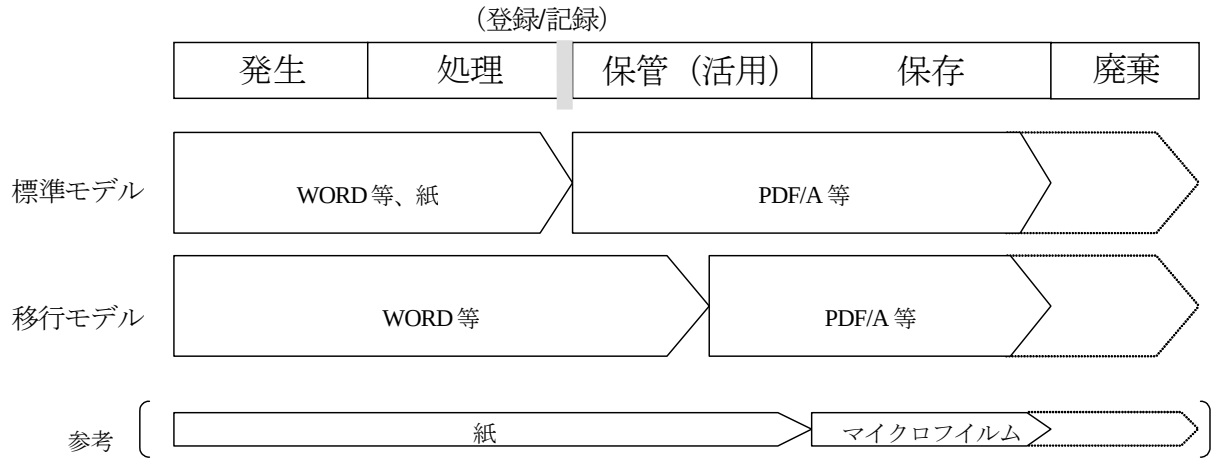


図 3.1: 電子文書のライフサイクルモデル（基本パターン）

記録に際して記録日時、記録者などを正しく残して行くことが重要である。この観点からすると、従来、長期署名フォーマットは長期保管の目的のみに利用すると受け取られがちであったが、JSOX 法で記録・保存が求められている内部統制関連文書のように保管期間が５年と比較的短期であっても、あるいは１年以下であっても、保管が必要ならば長期署名フォーマットを利用する価値がある。その理由を以下で解説する。

デジタル署名が有効であるためには、デジタル署名の生成者（署名者）の公開鍵証明書が有効である必要がある。公開鍵証明書の有効性は、有効期間と失効に左右される。

公開鍵証明書には予め有効期間が設定される。その期間は、特殊な場合を除けば数ヶ月から数年である。有効期間内に生成されたデジタル署名を有効期間内に検証した場合、有効と判断される。また、有効期間を過ぎてから生成されたデジタル署名は、無効と判断される。では、有効期間内に生成されたデジタル署名を、有効期間を過ぎてから検証する場合はどう判断すべきであろうか。この場合、「無効」と判断せざるを得ないのである。なぜならば、有効期間内に生成されたデジタル署名（A）と有効期間を過ぎてから生成されたデジタル署名（B）のデータを詳細に比較しても、どちらが有効期間内に生成されたものであるのか否かが区別できないからである。デジタル署名データ内に時刻情報を含めることは可能であるが、その時刻情報はデジタル署名を生成する PC のシステムクロックから取得したものであり、署名者による調整（偽装）が可能であるため、信頼することができない。たとえ、システムクロックが正確に同期を取られるよう管理されていたとしても、デジタル署名の時刻情報にそれが正しく反映していることを証明することは

困難である。(逆に署名した時刻や署名の有効性を証人や状況証拠などによって容易に証明できるのであれば、デジタル署名自体不要である。)

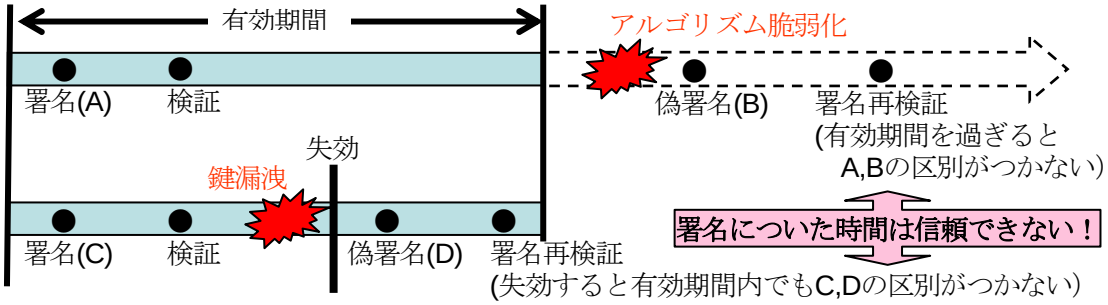


図 3.2: 公開鍵証明書の有効期間/失効とデジタル署名の有効性

失効の場合も同様である。一旦、失効が発生すると、デジタル署名が失効前に生成されたものであるか（デジタル署名（C））、失効の後に生成されたのか（デジタル署名（D））を判断することはできないため、全てのデジタル署名を「無効」と判断せざるを得なくなる。失効の場合、更に厄介なことに、有効期間を過ぎた後に公開鍵証明書が失効したか否かを示す失効情報を得ることが（通常は）できなくなってしまう。

このようなケースに備え、有効期間が過ぎても、失効が発生しても、更には失効情報を取得不可能になったとしても、当初そのデジタル署名が有効であったことを後日再確認（再検証）できるようにするのが、長期署名フォーマットである。

長期署名フォーマットは「長期」の語を冠してはいるが、決して１０年、２０年保存する場合にのみ有効な技術ではない。公開鍵証明書の有効期間が短い場合、あるいは有効期限が迫っている場合、その期限を越えて有効性を保つために、この技術を利用できる。失効を考えた場合、更にこの技術の価値が高まる。失効の要因としてはさまざまなケースが考えられ、しかも有効期間の場合と異なり、それがいつ発生するかわからない。つまり、受け取った時点では有効であったものが、記録として保管している間に（極端な場合、明日にでも）無効となってしまうことも考えられる。

すなわち、受け取ったデジタル署名付の文書の有効性をその時点で確認できればよく、後はデジタル署名を捨ててしまえるようなケース（これは記録とは呼べない）を除けば、その有効性を後々確認できるようにしておくことは必須である。すなわち、保管期限の長短は別にして、記録するという行為を行う場合は、長期署名フォーマットはその記録の信頼性を高めるための最有力で効果的手段である。

4. 電子保管媒体の動向

4.1. 光ディスク動向

（１）ＤＶＤ保存寿命の標準化

本 WG に対しテラバイト光メモリ研究開発機構（Tera Byte Optical Memory Consotium、以下 TBOC）殿から DVD の保存寿命の標準化の動向について紹介頂いた。

Writable DVD/CD については、メーカや型式の差により媒体の寿命は大きく異なっており、長期保存については、市場が混乱している。これに対し、米国の光ディスクの標準化団体である OSTA（Optical Storage Technology Association）と TBOC では、３０年の寿命を持つ媒体のテスト規格を策定し、ユーザの長期保存ニーズに答えようと活動している。

欧州電子計算機工業会(European Computer Manufacturer Association、以下 ECMA)により、２００６年6月２３日 ECMA 議案を上程し、ECMA 規格とした後、ISO 規格にしていく予定である。適合媒体には、ロゴマークも付与することを検討している。

これは、従来から ECOM が光ディスクメーカに要望していた事項であり、ECOM 要望が実現しつつある。

（２）ブルーレーザ光ディスク

市場に出回り始めたブルーレーザを使用した Blue-Ray、HD DVD は現在のところ民生対応を主としている。ビジネス用途向けに販売されていた Professional Disc for Data は販売中止になったこともあり、ビジネス用の保存媒体としてブルーレーザ光ディスクを採用するのには時期的に拙速の感がある。

4.2. 磁気テープのアーカイブへの適用

磁気テープの長期保存に関する課題については、前年報告では、参考文献「長期署名フォーマット相互運用性実験報告書」[3]の「2. 長期保存に使用時の磁気テープの課題」においてまとめた。本年は財団法人電子情報技術産業協会（JEITA）　磁気記録媒体標準化グループ殿の協力により、進展した事項について報告する。尚、以下の事項については本ＷＧとして次年度も引き続きその裏づけデータの入手などに務めていく予定である。

（1）　磁気テープの寿命

JEITA 殿から各テープメーカ等における実験、研究では磁気テープの寿命は大きく改善されているとの文献が紹介された。しかしながら、一般のテープ仕様として、寿命規定の記述がなく、第 3 者が測定できるような標準的な手法は未確立であった。各研究者、各製造会社などにより、異なった手法で測定しているのが実態であった。ECOM からは、第３者での測定手法の確立、測定結果の公開をお願いし、JEITA 殿の検討事項として頂いた。

（2）　定期的巻き直しの要否

JEITA 殿回答「最近のカートリッジ型テープ（LTO/IBM3592/DAT/STK 等）では不要です。」

（3）　耐塵埃性

仕様、実験データともににない。JEITA 殿からは「カートリッジ型テープは一般的に密閉型であり、通常の塵埃付着はない。」との回答を頂いたが、ECOM としては実験データが必要であるという見解である。

（4）　発塵埃性、テープ切れ・粘着性

JEITA 殿として、情報のまとめは行っていない。

今後も長期保存媒体として磁気テープを使用する場合は、各ユーザにおいて、使用する媒体、ドライブのメーカの両方が長期保存対応を約束することが必要と考える。

4.3. 保管／保存における電子媒体の選定基準に関する考察

磁気テープの長期保存性の検証、確認が期待される中、今後は、長期保存媒体としては磁気ディスク、光ディスク、磁気テープを並行して検討する時期に来ている。ついては、従来、システム運用設計上、誤解をされることが多かった事項について、本節で解説しておく。

文書のライフサイクル(図 3.1)に応じて、文書の発生日からの経過日数に応じて、一般的には、アクセス頻度（参照・更新）が図 4.3.1 のように減少するとされている。一方、磁気ディスクはアクセス速度が速いがビットコストが高い。一方、光ディスク、磁気テープなどの２次記憶装置はアクセス速度は遅いが、ビットコストが低いので、電子媒体のアクセス速度／コストの階層ピラミッド（階層構造）が図 4.3.2 のように示され、システム全体コストを下げるため、２次記憶装置は保管フェーズや保存フェーズに活用されてきた。

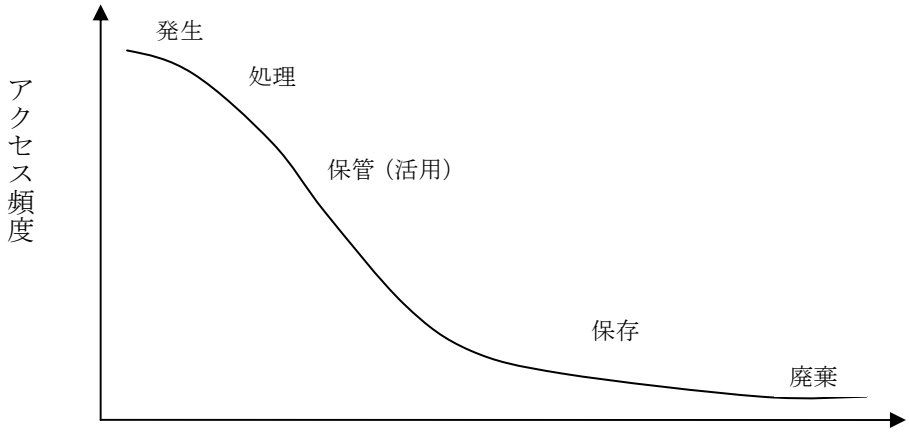


図 4.3.1: データ発生経過日数とアクセス頻度

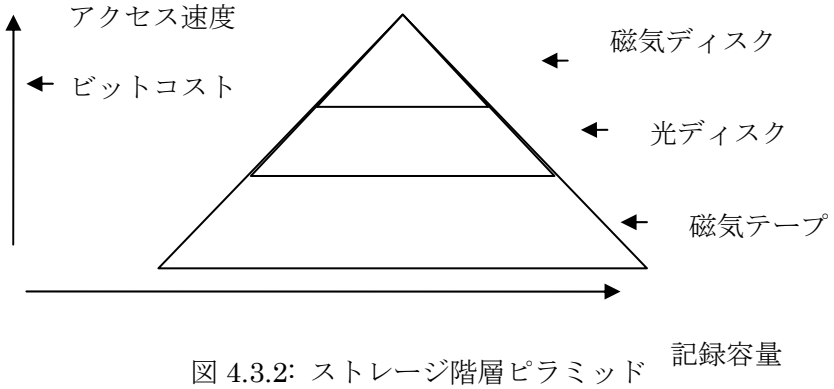


図 4.3.2: ストレージ階層ピラミッド

本節では、従来、システム運用設計上、誤解されることが多かった事項について解説する。

（1）許容参照時間

業務に使用される場合は、経過日数に応じて、参照頻度が低くなっても、参照に許される時間は、参照頻度によらず、業務内容、法規制、経過月数、経過年数に応じて決まるケースが多い。すなわち、要求されるサービスレベルに応じて、許容参照時間を決定し、これに応じて、記憶装置を選択することが望ましい。

（2）階層記憶管理システムにおけるチェンジャ製品利用上の留意点

磁気テープや、光ディスクを統合して、チェンジャ内に保管して、容量を拡大し、磁気

ディスクと組合せて、階層記憶管理システムを構成する場合、参照頻度の高いデータは自動的に、磁気ディスクに保存されており、高速に参照できる。参照頻度の低いデータは自動的にチェンジャから当該媒体を読出し装置に投入し、読出すとされている。したがって、単に、参照頻度の低いデータが多くあるという理由で、階層記憶管理システムを使用するのではなく、許容参照時間が決まっているケースにおいては、チェンジャ内の媒体からのデータ読出し時においても許容参照時間を満たすこと確認しておくことが必要である。

（３）チェンジャへの同時アクセス

光ディスク、磁気テープなどの電子媒体を利用したチェンジャを使用した場合、チェンジャ内の電子媒体を運ぶアクセッサは通常、装置毎に、１台、多くて２台となる。すなわち、参照頻度が少なくても、アクセスが重なれば、前の処理が終了するまで、アクセッサが次の媒体を交換できなくなる。通常、チェンジャ内での電子媒体の交換には３０秒から１分程度の時間を要する。したがって、参照許容時間を決められているケースにおいては、使用ユーザ数を限定する必要がある。

（２） 参照ファイル容量による媒体の選定

光ディスクと磁気テープを比較すると、アクセス速度については「光ディスク」が優位、データ転送速度については、「磁気テープ」が優位となっている。従って、保管媒体を選定する場合は、保管ファイルの容量も検討材料とし、小容量では光ディスクが適する、大容量には磁気テープが適するということも考慮するのが望ましい。

参考文献

[1] 「財務報告に係わる内部統制の評価及び監査に関する実施基準」－公開草案－, 平成 18 年 11 月 21 日, 金融庁 企業会計審議会内部統制部会

[2] 「システム管理基準 追補版(財務報告に係わる IT 統制ガイダンス」(案), 平成 19 年 1 月 19 日, 経済産業省

[3] 平成 16 年度 ECOM 報告書「電子文書の長期保存と見読性に関するガイドライン」

[4] 平成 17 年度 ECOM 報告書「長期署名フォーマット相互運用性実験報告書」

[5] JIS X 0902-1 「情報及びドキュメンテーション-記録管理- 第 1 部:総説」

[6] ISO 15489-1 Information and documentation-Records Management Part1 General

[7] ISO 15489-2 Information and documentation-Records Management Part2 Guidelines

[8]「財務報告に係わる内部統制の評価及び監査に関する実施基準」、平成 19 年 2 月 15 日、金融庁企業会計審議会内部統制部会

第三部

長期署名プロファイルの JIS 化、
相互運用実験、及び関連する国際調整

まえがき

第三部では、今年度に作業部会で策定した長期署名プロファイルの JIS 原案の概要、長期署名プロファイルに基づいた製品の国内及び欧州との相互運用実験の準備状況、並びに、ETSI/TC ESI や ISO TC171/SC2/WG5 との長期署名の仕様に関する国際調整の状況に関して述べる。

1. 長期署名プロファイルの JIS 化

1.1 JIS 化の経緯

電子文書の真正性を担保する電子署名やタイムスタンプは、その多くが公開鍵基盤技術を利用しており、公開鍵証明書の有効期限を越えて署名の検証ができない。この問題を解決する技術の一つに、公開鍵証明書の有効期限が切れる前に新たなタイムスタンプを付与する“長期署名”技術がある。

ECOM は 2000 年度から継続的に“長期署名”技術の調査研究に取り組んでおり、昨年度には、欧州通信規格協会(ETSI)が規定した TS 101 733 (CMS Advanced Electronic Signatures、CAdES)及び TS 101 903 (XML Advanced Electronic Signatures、XAdES)をベースとした“CAdES 長期署名プロファイル(Version1.0)” と“XAdES 長期署名プロファイル(Version1.0)”を策定した。これらは昨年度の報告書[1]の付録として掲載されている。

“長期署名プロファイル”は、“長期署名”の観点から、CAdES や XAdES の構成要素に対する要求レベル（必須、任意選択など） や曖昧な記述の解釈方法を定義している。また、“長期署名プロファイル”の適用領域は、一般文書、契約書類、署名付きの電子メール、及びダウンロードファイルなどにおける、長期署名の設計や実装を想定している（図 1.1.1)。署名対象ファイルは、PDF/A ファイル、XML ファイル、TIFF ファイルなどである。

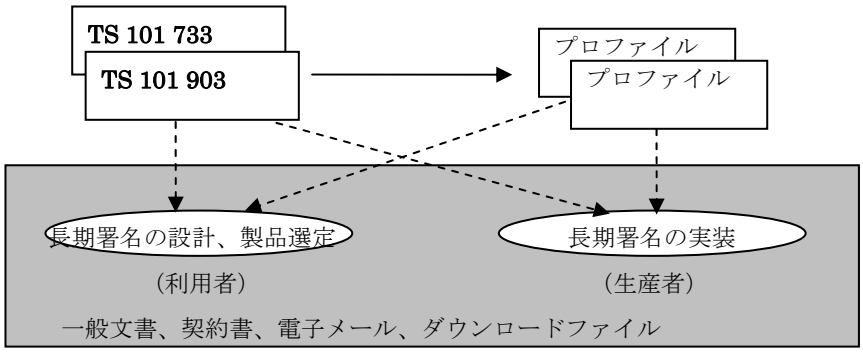


図 1. 1. 1: 長期署名プロファイルの適用範囲

今年度は、その普及活動の一環として、生産者には相互運用性の確保された実装を行うための指針を、また、利用者には、長期署名の方式設計や、相互運用性の確保された製品選択の指針を提供すべく、JIS 化に向けての原案作成に取り組んだ。これにより、電子署名やタイムスタンプが付与された文書の更なる相互運用を可能にし、電子文書の流通、活用、ならびに長期にわたる保存が可能になる。

“長期署名プロファイル”の JIS 原案（本報告書の付録参照）作成に当たっては、昨年度 ECOM が策定した“長期署名プロファイル”（以降、区別のために ECOM プロファイルという）を基礎としたが、CAdES や XAdES の曖昧な記述に対しては独自の解釈を与えるのではなく、参照する CAdES や XAdES の標準そのものの品質を上げるアプローチを採用した。2006 年 3 月に、ECOM は、CAdES や XAdES の仕様策定を行っている欧州通信規格協会 電子署名基盤 技術委員会 (ETSI TC ESI: European Telecommunication Standards Institute, Technical Committee on Electronic

Signatures and Infrastructures、以下 ETSI TC ESI)と非公式ながらリエゾン関係を結び、CAdES V1.6.3 を参照しながら、バージョン間の後方非互換の解消や、複数解釈が可能な規定部分に関する規定の明確化に向けて調整を実施した。主な調整結果は次のようなものである。

a) バージョン間の後方非互換性に関しては、(当然の事であるが)処理が異なる属性には別のオブジェクト識別子を付与する。

b) アーカイブタイムスタンプのハッシュ演算では、事前に識別符号化規則(DER)処理を行わない。

c) 外部署名のためコンテント(eContent)が省略されている場合も、外部のコンテント(eContent)をアーカイブタイムスタンプのハッシュ演算対象に加える。

d) カウンタ署名(Countersignature)に対して、個別のアーカイブスタンプを付与しない。

これらの調整結果は、CAdES の最新バージョン(v1.7.3)に反映された。交渉の詳細は 2 章に述べる。

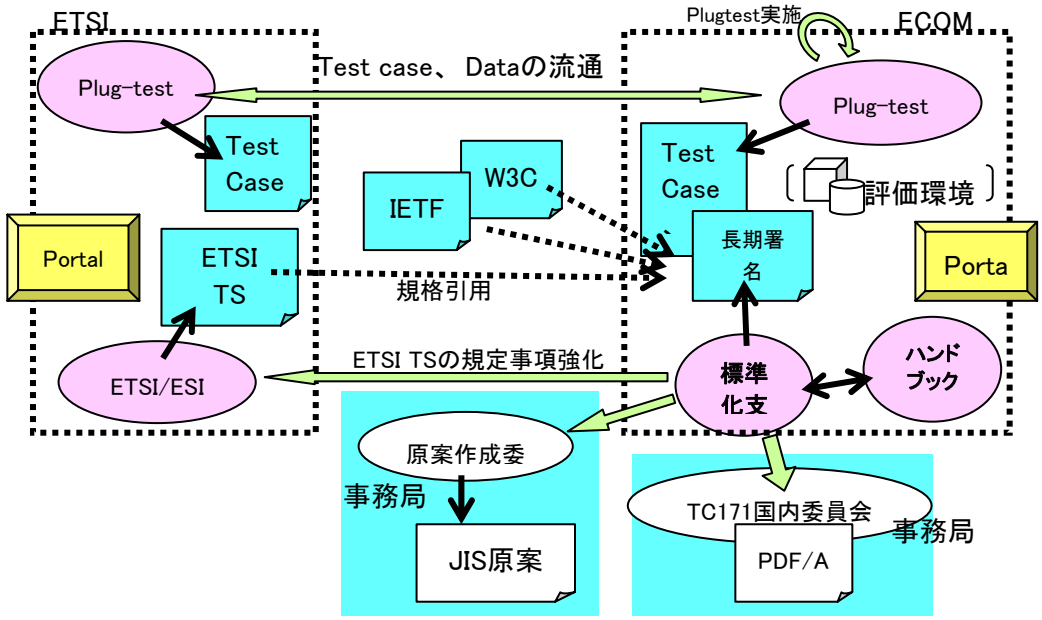


図 1.1.2: 長期署名プロファイルの JIS 化推進体制の全体像

1.2 JIS 原案の概要

1.2.1 用語及び要素名

JIS 規格で使われる用語や要素名は日本語が基本になる。このため、JIS 化原案作成に当たり、まず、引用規定(CMS、XML 署名、CAdES、XAdES)で使われている用語並びに要素名の日本語名称の選定と評価を行った。

日本語名称に関しては、分野の近い他の JIS 規格を参考にしながらこれまで慣用的に用いられているものは原則そのまま採用し、新たな名称が必要となるものについては、表 1.2.1.1 の原則に基づいて付与した。

表 1.2.1.1: 用語及び要素名の対応和文に関する原則

英文	和文
signature〜	署名〜
signer〜	署名者〜
signed〜	署名〜
unsigned〜	非署名(の)〜
content〜	コンテント〜
〜certificate	〜証明書
digest〜	ダイジェスト〜
〜reference	〜参照情報
〜value	〜値
〜identifier	〜識別子
〜s	〜群
〜version	の版数
other〜	他の
〜element	〜要素
〜attribute	〜属性
〜property	〜プロパティ
qualifying〜	〜を特定する

1.2.2 プロファイリング

CAdES や XAdES では、フォーマットとして、基本署名(CAdES-BES/ XAdES-BES)、署名ポリシ明示署名(CAdES-EPES/ XAdES-EPES)、時刻付き署名(CAdES-T/XAdES-T)、認証経路を構成する全証明書照合情報付き署名(CAdES-C/ XAdES-C)、全検証情報付き署名(CAdES-X/ XAdES-X)、長期保存署名(CAdES-A/ XAdES-A)が定義され、これらが包含関係にある。しかしながら、実際には、図 1.2.2.1 のように、文書への署名及びタイムスタンプの付与を行った CAdES-T/XAdES-T データによる流通・保管と、アーカイブタイムスタンプを付与した CAdES-A/XAdES-A データによる保存という運用形態が主で、これ以外の中間的な状態で流通・保管、保存されることは考え難く、製品も極めて稀であると想定されることから、JIS 化に当たっては、次の、時刻付き署名(CAdES-T/XAdES-T)と長期保存署名(CAdES-A/XAdES-A)を対象にプロファイリングを行った。

a)時刻付き署名(CAdES with time、CAdES-T/XAdES with time、XAdES-T)プロファイル
信頼のおける署名時刻（例えば、署名タイムスタンプ）を伴う署名データの生成及び検証に関するプロファイル。

b)長期保存署名(Archival CAdES、CAdES-A/Archival XAdES、XAdES-A)プロファイル
署名対象及び検証情報を含む署名に関する情報の改ざん(竄)検知が可能な措置（例えば、アーカイブタイムスタンプ）を伴う署名データの生成及び検証に関するプロファイル。

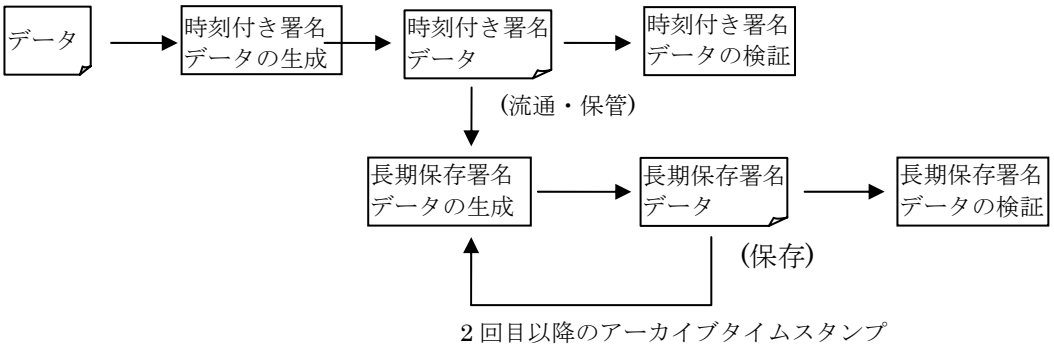


図 1.2.2.1: 時刻付き署名データと長期保存署名データの関係

1.2.3 要求レベル

一般に、規格の**“任意選択”**要素は、相互運用性を阻害する大きな要因となる。“長期署名プロファイル”では、引用規格の**“任意選択”**要素に対し、プロファイルとして、**“必す(須)”**、**“任意選択”**、**“要別途規定”**の 3 つの要求レベルを定義することにより、相互運用性を確保する。

“要別途規定”は、これに該当する要素の実装を排除するものではない。目的別に曖昧性のない仕様が定義され広く開示されることを期待している。

“必す(須)”: この要求レベルをもつ要素は必ず実装しなければならない。この要求レベルの要素が、選択肢となる下位要素をもつ場合は、少なくともその一つを選択しなければならない。
“任意選択”: この要求レベルを持つ要素の実装は任意である。
“要別途規定”: この要求レベルを持つ要素の実装は任意であるが、その処理に関して、別途に詳細な仕様を規定しなければならない。

CAdES データを構成する各要素に対する要求レベルは、次の基準に基づいて設定した。

a) 暗号メッセージ構文を利用した電子署名(CAdES)の定義で必す（須）となっている要素、並びに長期署名の生成及び検証に最低限必要な要素は**“必須”**とする。
b) 外部定義要素は、**“要別途規定”**とする。
　例 その他形式の証明書(OtherCertificateFormat)
c) 特定アプリケーションとの連携を前提とした要素は**“要別途規定”**とする。
　例 コンテント参照情報(ContentReference)
d) 運用に依存する要因が残っている要素は、**“要別途規定”**とする。
　例 属性証明書系要素、タイムマーク(TimeMark)など

XAdES データを構成する各要素に対する要求レベルの基準も同様である。

注記 **ISO/IEC 1804-2** で定義されたアーカイビング方式のタイムスタンプ(Time-stamps using

archiving)はその他タイムスタンプに含まれる。

e) 単なる参考情報を運ぶ要素に関しては、その要素を実装していない場合、要求を無視する。

例 署名時刻(SigningTime)

1.2.4 CAdES における ECOM プロファイルとの互換性

ECOM プロファイルは、TS 101 733 v1.6.3 を対象としているが、アーカイブタイムスタンプの定義の曖昧性を排除するために、アーカイブタイムスタンプ要素以外は v1.6.3 を参照し、アーカイブタイムスタンプ要素のみ v1.4.0 を参照している。このため、“長期署名プロファイル”では、ECOM プロファイル実装製品との相互運用性を確保すべく、v1.4.0 のアーカイブタイムスタンプ要素を**“任意選択”**として選択肢に加えた。

1.2.5 附属書

附属書(参考)として添付した、“PDF/A への長期署名の適用方法”は、PDF/A 仕様に従ったファイルフォーマットに対する長期署名データの組み込み方の推奨案である。定義内容が、PDF/A の規格に反映された場合は、この**附属書**は削除する予定である

備考：本件に関して、昨年度の報告書では幾つかの案を提示した。今年度は更に検討を進め、第５の案である、添付ファイルに長期署名データを格納する方法を推奨案とすることとした。

1.3 JIS 化委員会での主な審議事項

JIS 原案は、長期署名 JIS 化原案作成委員会（委員長：辻　秀一　東海大学教授）及び長期署名 JIS 化原案作成作業部会においいて作成・審議されれた。JIS 化原案作成委員会での主な論点と審議結果は次の通りである。

(1)全体構成とタイトル

JIS 原案の構成は、CAdES 版と XAdES 版の二分冊とした。用語定義など重複するものに関してはいずれか一方に定義し、もう一方はそれを参照するのが通常の構成であるが、使い勝手を考えて、一冊で完結するようにした。また、プロファイルという用語は一般に馴染みがなく、タイトルを見て規定内容を推測することが困難であるとの指摘があったことから、プロファイルは要求事項の意味も含むが敢えてプロファイルに関する要求事項とした。

(2)引用規格

JIS 作成ガイドライン[2]に、引用規格は、ISO、IEC、JIS またはそれに順ずるものとの規定がある。このため、CAdES や XAdES を引用規格ではなく“引用規定”として位置付け、引用規格のなかに掲載した。また、用語定義や CMS の抜粋など引用可能な JIS 候補として JIS X 5063-1 タイムスタンプサービスの枠組みがあったが、引用している CMS 構文が 2 世代古く、引用規定と不整合が発生する恐れもあることから、これを引用せずに再掲した。

備考：規格内の“引用規定”の定義方法に関しては，その扱いが過渡期にあり流動的である。

(3)規格適合性

規格への適合性を確かめることができる具体的な方法を記載する必要があるとの指摘があり、附属書として、プロファイル適合性宣言のためのチェックリストを添付した。

(4)要求レベル

必須、任意選択、要別途規定の関係の明確化が求められ、次のように整理した。

- 十分定義されている
  - 必須
  - 任意選択
- 定義が十分ではない（要別途規定）

(5)用語

用語については、Advanced Electronic Signature 、Signed Data、Encapsulated Contents、Counter Signature、Commitment、～s（複数形）などの和訳に対して大小さまざまな意見や指摘があった。これらを含め審議結果を本報告書の付録に掲載する。ここでは、委員会で多くの時間を割いて議論された Advanced Electronic Signature と Counter Signature について論点と結果を記す。

a) Advanced Electronic Signature

CAdES/XAdES のタイトルの一部である Advanced Electronic Signature は EU における電子署名のレベルを反映したものであるが、国内にはこれに該当する電子署名のレベルの概念がなく、また、この規格では CAdES/XAdES のフォーマットのみを参照していることから、 Advanced Electronic Signature の Advanced を特に訳さず、単に電子署名とした。

b) Counter Signature

CAdES/XAdES のフォーマットを構成する要素の一つとつとして Countersignature 要素が定義されているが、この要素は、順序に意味を持つ署名に使われることを想定しており、必ずしも副署の意味では使われないこと、順序に意味を持つという構造的な観点からは、直列署名や連鎖署名という名称が候補となるが、この構造は，例えば CAdES では，署名付きデータ(SignedData)のネスト構造で実現することも可能であることから、順序のみに着目し署名に対する署名に限定した日本語名称として、カウンタ署名とした。

2. 長期署名製品の相互運用性実証実験

昨年度(2005 度)、ECOM では CAdES もしくは XAdES の実装を有する企業、並びに協力企業 14 社が参加し、同年策定した ECOM 長期署名フォーマットプロファイル 2005 への実装の準拠性を確認するための相互運用性実証実験を行った。今年度も、2006 度末から 2007 度にかけて、以下の実証実験を行う計画である。

- 策定された JIS 標準への準拠性を確認するための国内相互運用性実証実験
- JIS プロファイルが欧州や他国との相互運用性を持っていることを確認するための国際相互運用性実証実験

2.1 JIS 準拠性に関する実証実験

2006 年 12 月 4 日に、“JIS 原案による長期署名フォーマットプロファイルの相互運用性テスト参加企業の募集開始”をプレスリリースした。JIS 原案のプロファイルに基づいたテスト仕様を作成し、参加各社の製品の相互運用性テストを 2007 年 3 月までに行う予定である。また、ETSI TC ESI とテスト計画を調整し、欧州企業と日本企業が参加する相互運用性テストを 2007 年 10 月頃に行う計画を進めている。

2.1.1 CAdES 実証実験

以下の内容で実証実験を実施する。

- CAdES v1.4.0～v1.5.1 で ECOM プロファイル 2005 に基づいているならば、JIS 準拠であり、2005 度作成したテストデータがそのまま準拠性評価のために再利用できるため、本年度テストの主目的とはしない。
- CAdES v1.7.3 に基づく製品であれば、データ構造が JIS で規定された領域に従っているならば JIS 準拠である。本年度の実験では、v1.7.3 の追加変更された要件である以下のテストケースの追加実験を行う。
  - ・署名者証明書情報での ESSCertV2 属性の対応
  - ・ArchiveTimeStampV2 属性
  - ・昨年度は行わなかった CounterSignature 属性

2.1.2 XAdES 実証実験

ECOM プロファイル 2005 はドラフト版 v1.3.1 に基づいていたが、JIS では最新版の XAdES v1.3.2(以降)に基づいている。全てのテスト項目についてデータを再生成し、テストを実施する。

2.2 欧州との実証実験

以下、ETSI との実証実験の準備状況を時系列にて報告する。

(1) ETSI-ECOM 実証実験の打診（2006 年 9 月）

ETSI TC ESI の Dionisio Zumerle 氏より、これまでの ETSI TC ESI に対する ECOM の貢献につ

いて謝辞を頂いたと共に、以下の内容案で ETSI-ECOM 実証実験の招待を受けた。

- テスト対象：XAdES、 CAdES
- 期間：2007.1Q から 2007.4Q もしくは 2008.1Q まで
- 場所：未定

XAdES については以下の検討が必要であるとした。

- テスト範囲：XAdES のみか、OASIS DSS TC などの範囲も含めるか
- テスト方法：対面かリモートか
- 参加費用はどうするか

  日本回答：参加費は無料としたい
- テスト対象のバージョンは：XAdES v1.3.2 でよいか

詳細な議論は ESI#15 会議(2006 年 10 月 31 日、11 月 1 日)で行うこととした。

(2) ETSI TC ESI#15 ベルリン会議(2006 年 10 月 31 日～11 月 1 日)

以下の方向で調整を進めることで合意した。

- 基本的に対面ではなくリモートテストとする。
- 担当者による対面会議を数回実施する。
- テスト参加費は無料とする。
- テストに関する技術的な詳細は担当者が電話会議で詰める。

(3) 第一回電話会議（2006 年 11 月 6 日 日本時間 17:00-18:00）

ECOM 側からは以下の提案を行った。

- CAdES、 XAdES 両者をテスト対象とする。
- データの生成と検証の双方のテストを対象とする。
- ECOM 側の現状の参加組織は CAdES が 15、XAdES が 5 程度を予定
- リモートテストが望ましい
- 必要に応じ対面会議を 3 回程度行ってはどうか
- ツールによる鍵、証明書、CRL、署名テストデータの提供が可能
- テスト用 TSA 局については、日本からの提供も調整可能
- スケジュール（2007 年 1 月～3 月:テスト設計 2007 年 7 月～1 月実施）

電話会議では以下の内容で合意した。

- CAdES、 XAdES の参加組織のリストを 2006 年 12 月末までに提示する。
- 2007.1 月～3 月で事前テストを行うこととする。

電話会議においては、ETSI の CAdES の標準に携わる中心メンバが参加していないため、当面は XAdES 実証実験主導で進めることとなる。

(4) 第二回電話会議（2006 年 11 月 28 日 日本時間 17:00-18:00）

これまでの議論により、XAdES 相互運用性テストを先行させて行うこととなった。2007 年、実証実験を行うにあたり、事前に ETSI TC ESI および ECOM のコアメンバで事前にテストを行い、予めテスト内容について詳細を詰めておくこととした。ECOM 側で事前テストのための確認項目をまとめ、確認を行った。

- 事前テスト参加組織
  カタルーニャ工科大(スペイン)、グラッツ工科大学(オーストリア)、日本電気、エントラストジャパンの 4 組織とする。
- テストする標準
  ETSI TS 101 903 v1.3.2 XAdES とする。
- 必要なサービスの提供
  TSA については日本で調整、CA はオーストリアとする。
- CA 信頼モデルについて
  階層モデルとする。
- 今後、テスト設計書の作成に着手する。雛形はスペインで提供する。

(5) XAdES 事前テスト 設計書執筆開始（2006 年 12 月 22 日）

以降、以下のタイムテーブルで執筆が進められた。

- 2006 年 1 月 11 日 第一版ドラフト回覧
- 2006 年 1 月 14 日 目的、テスト TSA を追記、テストの必須・オプションの区別
- 2006 年 1 月 25 日頃 最終稿予定

3. 長期署名仕様に関する国際調整

今年度は、長期署名の仕様に関する国際調整に進展があった。以下では、ETSI TC ESI との連携による長期署名仕様(ETSI TS 101 733 及び TS101 903)の見直し、及び、ISO TC171/SC2/WG5 への PDF/A(ISO 19005)に対する長期署名の組み込み提案の結果について報告する。

3.1 ETSI TC ESI との仕様調整

ETSI TC ESI は、欧州における長期署名フォーマット、タイムスタンプおよび署名ポリシの標準を策定し、長期署名フォーマットに関しては事実上の国際標準となっている。

これまで、ECOM 公証認証 WG、セキュリティ WG では 2000 年より長期保管、長期署名フォーマット、タイムスタンプについて調査を行っており、以降、ETSI TC ESI との関係を徐々に築き上げ、現在、標準改訂のレビューや相互運用実験を行うまでの協調関係を結んでいる。

本年度、ECOM は ETSI TC ESI に向けて以下の活動を行った。

- ETSI TC ESI Meeting への参加
- CAdES において、技術的な曖昧性を排除する仕様への反映
- ETSI-ECOM CAdES/XAdES 相互運用性テストの準備

ちなみに、ETSI のメンバーは CAdES を「キャデス」、XAdES を「シャデス」と発音している。

3.1.1 欧州の電子署名標準化の現状

EU における電子署名を定めた EU 電子署名指令(Directive 1999/93/EC on Electronic Signature)を受けて、幾つかの組織が設立されてきた。EU 指令を推進する中核的な組織であった 1999 年に設立された欧州電子署名標準化イニシアチブ(European Electronic Signature Standardization Initiative、以下 EESSI)は、その目的を達成したとして 2004 年に終了した。また、CEN/ISSS E-Sign ワークショップは 2003 年に同様に終了した。

2007 年現在、欧州において電子署名に係る新しい標準の策定や既存の標準の保守は ETSI TC ESI 主体で行われている。

3.1.2 CAdES 仕様の調整

本節では、今年度の CAdES の標準に関する ETSI に対する ECOM の活動について、時系列にて報告する。標準や標準案に対して継続してコメントし続け、大元となる標準を改定してもらうことで、CAdES 標準の処理や解釈の曖昧性を排除し、日本独自の解釈を加えることなく国際相互運用性を確保できるようになった。

(1) ETSI TC ESI #13 バルセロナ総会へのオブザーバー参加(2006 年 3 月 21 日～22 日)

ETSI TC ESI#13 会議に参加し、ECOM における CAdES、 XAdES に関する調査研究、普及活動

について報告を行い 2005 年に行われた ECOM 長期署名フォーマットプラグテストについて報告を行った。また、現状認識している CAdES、XAdES の仕様の相互運用性上の問題点について指摘を行い、ETSI TC ESI と ECOM が意見交換を行うことが合意された。

(2) CAdES の仕様に関する意見交換(2006 年 3 月下旬)

CAdES の仕様策定の担当者(Nick Pope 氏)と ECOM が指摘した仕様の問題点についてメールベースで意見交換を行った。

- 問題点 1: v1.4.0 と v1.5.1 ではアーカイブタイムスタンプのハッシュの計算法が異なるが、個別 OID を振り分けなかったことにより、相互運用性、後方互換性の問題がある。
  回答 1: v1.5.1 に新しい OID を付与し損ない、処理の曖昧性を残してしまったことを認識している。これを解決するのは課題とする。
- 問題点 2: 古い v1.4.0(or RFC 3126)の方がアーカイブタイムスタンプのハッシュ対象の計算法が厳密であり、相互運用性の課題も少ない。v1.5.1 では、どのように生成、検証すべきか記述が曖昧すぎる。
  回答 2: v1.5.1 では処理方法が曖昧になったことは認識している。但し、v1.5.1 では全ての非署名属性が対象となり v1.4.0 では対象となっていなかった CounterSignature 属性なども保護することができる。v1.4.0 に戻ることはなく v1.5.1 以降の方法を推奨する。
- 問題点 3: タイムスタンプトークンを検証するための CRL などの検証情報をどこに格納するのか明示的でない。
  回答 3: 説明文を追加することを検討する。
- 問題点 4: CAdES のエンコーディングの DER/BER の違いについて、CMS で記述されているようには eContent や非署名属性について明確に記述されていない。この問題は特に v1.5.1 のアーカイブタイムスタンプ属性の場合には、問題となる。
  回答 4: 今後検討する。非署名属性および他のハッシュ対象データも DER でエンコードされるべき。 但し、オリジナルデータのエンコーディングが保持され続けるならば、 DER エンコードでなくてもよい。 eContent は DER である必要はない。

また、現状 ETSI TC ESI で追加検討されている、TS 101 733 v1.7.3 に向けた改正点についての情報が送られてきた。

- 改正点 1: 署名ポリシ識別子の SignaturePolicyId において、SignaturePolicyId オプショナルとするのを止め、sigPolicyHash の値の長さが 0 ならば、「ハッシュ値は未知のポリシ」と解釈することとする。
  ECOM 側コメント: 値長さ 0 なら、ハッシュ値不明とするならば、ASN.1 による定義の通例上 sigPolicyHash をオプショナルとするのが、通例ではないか。
- 改正点 2: OtherSigningCertificate を非推奨とし、GeneralSigningCertificate を使用することとする。

ECOM 側コメント: IETF S/MIME WG で提案されている ESSCertV2 を使うことを検討してはどうか。同じ目的、同じ構造の ASN.1 クラスが幾つも定義されることは問題。

(3) CAdES の仕様に関する意見交換 2(2006 年 4 月上旬)

ECOM 側から現状の CAdES v1.5.1 以降の問題点について整理し ETSI TC ESI に提示した。これに対し ETSI TC ESI より予想通りの回答を得た。

- 問題点 1: 古い v1.4.0 でも新しい v1.5.1 以降でも分離署名の場合、eContent 内には署名対象データは無いので、アーカイブタイムスタンプは署名対象データを保護しない。
  回答 1: 分離署名については別途方法を検討する必要がある。
- 問題点 2: 内包署名におけるアーカイブタイムスタンプハッシュ値計算の際、署名対象データの情報 eContent が、ASN.1 BER の不定長表現(indefinite length representation)であるか、ASN.1 DER の固定長表現であるかによってハッシュ値が異なってしまう。CMS SignedData の検証時に signedAttributes の DER 化を行うような何らかの正規化ステップを踏むか、それとも、入力と出力とで値を変えずにそのまま計算するかを明確にしなければならない。
  回答 2: 正規化処理を行わず「そのまま(AS IS)」をハッシュ対象に加えることを推奨する。
- 問題点 3: アーカイブタイムスタンプでは signerInfo の全ての要素をハッシュ対象に加える。その際、ASN.1 DER で正規化することにより BER の不定長表現、および、SET OF コンポーネントのソートなどの揺らぎの問題を解消するのか、それとも「そのまま(AS IS)」をハッシュ対象に加えるのかを明示しなければならない。
  回答 3: 正規化処理を行わず「そのまま(AS IS)」をハッシュ対象に加えることを推奨する。
- 問題点 4: アーカイブタイムスタンプにおいて unsignedAttributes 領域をハッシュ対象にする際、SET OF コンポーネントのソートを行うのか、それともそのまま(AS IS)でハッシュ対象に加えるのかを明らかにしなければならない。
  回答 4: 正規化処理を行わず「そのまま(AS IS)」をハッシュ対象に加えることを推奨する。
- 問題点 5: 何世代かのアーカイブタイムスタンプを追加による署名延長を行う間、アーカイブタイムスタンプ以外の属性を追加することは可能なのか。その場合の処理はどのようになるのか明示しなければならない。
  回答 5: アーカイブタイムスタンプを追加したら、それ以降は他の種類の属性を追加することを禁ずる。
- 問題点 6: CAdES では CMS SignedData をバージョン 3 に限定している。この場合、属性証明書を certificates 領域に入れるか、signerInfo のバージョンを 3 とする必要がある。signerInfo をバージョン 3 にするためには、IssuerAndSerial ではなく subjectKeyIdentifier の値を指定する必要があるため署名者証明書には subjectKeyIdentifier が必要となる。このような制限は不要ではないか。
  回答 6: 制限はしていない。(備考：この時点では CMS SignedData のバージョン判定に

対する ETSI TC ESI の認識が薄く ECOM の主張する問題点を理解してもらえなかったが、最終ドラフトではバージョン 3 の制限は解除された。)

(4) CAdES v1.6.3 改定案についての議論 (2006 年 6 月中旬)

ETSI TC ESI より ArchiveTimeStamp 属性に関する仕様変更案に対するコメントを求められた。これまで ECOM 側から指摘していた ArchiveTimeStamp のハッシュ計算に対する v1.5.1 以降の曖昧性を排除するための補足説明が多く追加されていた。改定案における変更点を以下に示す。

- 改定案 1(修正): id-aa-ets-archiveTimeStamp を別 OID(v2)とする。OID はこの時点では未定。
- 改定案 2(加筆): encapContentInfo の eContent が省略された分離署名の場合、署名値により eContent を保護する。
- 改定案 3(加筆): v1.5.1 より前の OID(1.2.840.113549.1.9.16.2.27)のアーカイブタイムスタンプは v1.5.1 以降とは互換性が無い。v1.5.1 と v1.6.3 は内包署名の場合には互換性がある。署名検証者の全てが合意している場合を除き、旧アーカイブタイムスタンプは非推奨とする。
- 改定案 4(加筆): SignedData 構造に含まれているので、CounterSignatures 属性については、それに対し別途タイムスタンプを必要としない。(備考: SignerInfo 構造の unsignedAttributes に含まれるので、アーカイブタイムスタンプで保護されている。)
- 改定案 5(加筆): アーカイブタイムスタンプの対象となる ASN.1 構造は「そのまま(AS IS)」でハッシュ対象に加えられる。
- 改定案 6(加筆): ハッシュ対象は ASN.1 構造の型(T)と長さ(L)を含む要素を繋げたデータである。
- 改定案 7(加筆): 事前合意無しには、unsignedAttributes が DER エンコーディングであるとは保証できない。

(5) ETSI TC ESI#15 ベルリン総会へのオブザーバー参加(2006 年 10 月 31 日～11 月 1 日)

総会では CAdES v1.7.3 に向けた改定に関する現状報告があった。ECOM より、ETSI と ECOM とで CAdES、XAdES に関する相互運用実証実験を行いたい旨の提案を行い、了承された。総会以降、両組織担当者による電話会議により、スケジュール、実験内容、参加方法などを詰めることとした。

会議では ETSI 会員でない ECOM からのコメントを汲む必要があるのかという意見もあった。(備考：2006 年末時点で、ECOM は 2007 年より ETSI の正会員になるよう調整を進めている。)

(6) CAdES v1.7.3 ドラフト RevC に対するコメント(2006 年 11 月上旬)

ETSI TC ESI から CAdES v1.7.3 のドラフト RevC が送付され、コメントを求められた。ドラフトではかなり多くの部分において、これまでの ECOM からの提言が盛り込まれた形になっていた。以下の項目についてコメントを送付し、ETSI TC ESI より回答を得た。

- 提案 1: CMS SignedData type をバージョン 3 に制限する理由が無い。
  回答 1: 承認する。v3 という制限を除く。
- 提案 2: SignaturePolicyId の sigPolicyhash が値の長さが 0 である場合に特別な解釈を与えるならば、OPTIONAL とするべきではないか。
  回答 2: 後方互換性から提案を却下する。
- 提案 3: CrlIdentifier において crlIssuedTime は GeneralizedTime もあるため、UTCTime に制限せず Time 型とすべき。
  回答 3: 基本的に承認する。RFC3280 の CRL の thisUpdate、nextUpdate は UTCTime のみなので 2049 年までは必要ない。CAdES の RFC と v1.7.3 の後の版でこれを盛り込む。
- 提案 4: CrlValidatedID と OcspResponsesID においては hash と identifier の必須、オプションの関係が逆となっている。一貫させるべきではないか。
  回答 4: 後方互換性から提案を却下する。
- 提案 5: タイプミスの指摘。変更履歴補足の指摘。
  回答 5: 承認する。

仕様のリリースのスケジュールについては、TS 101 733 v1.7.3 は 1 月か 2 月にリリース予定。CAdES の IETF RFC は、その後となるとの回答を得た。

(7) CAdES v1.7.3 の ArchiveTimeStampV2 のハッシュ対象(2006 年 11 月中旬)

v1.7.3 の改定案がほぼ収束に向かっており、最終的に ArchiveTimeStampV2 のハッシュ計算対象について ETSI TC ESI と ECOM とで齟齬が無いか確認のため、計算方法を解説するドキュメントが送付されてきた。一部誤解を招く表現があり、コメントを送ったところ、修正された。

曖昧さを含み誤解される恐れのある点を ECOM にて以下のようにまとめた。

- **外部署名について**
  外部署名の場合には、対象をそのままハッシュ対象に加えるのであり、encapContentInfo 構造の中の eContent 中に外部署名対象を含めるような構造を再生成することではない。
- **非署名属性群について**
  ArchiveTimeStampV2 の検証の際には、ハッシュ対象の unsignedAttributes の再構築が必要である。検証対象の ArchiveTimeStampV2 の時刻以降の ArchiveTimeStamp を除いて、他の属性の内容は構造的に決して変更が無いように注意したまま、各属性の順序をソートしたりしない BER “[1] IMPLICIT SET OF” 構造を構築し、これをアーカイブハッシュ対象とする。
- **署名属性群について**
  ArchiveTimeStampV2 の検証の際には、CMS SignedData 署名のハッシュ対象の処理と異なりハッシュ対象の signedAttributes は、"[0] IMPLICIT SET OF"を"SET OF" に直したり、BER を DER に変換するなどの正規化処理は必要とせず、そのまま(AS IS)をアーカ

イブハッシュ対象に加える。

(8)　CAdES v1.7.3 ArchiveTimeStampV2 属性の OID (2007 年 1 月)

ArchiveTimeStampV2 属性の OID が未定となっていたが、ETSI TC ESI に問い合わせたところ、IETF S/MIME WG との調整により以下の OID (1.2.840.113549.1.9.16.2.**48**) を採用することとなったとの回答を得た。

```
aid-smime OBJECT IDENTIFIER ::= { iso(1) member-body(2)
                                  us(840) rsadsi(113549) pkcs(1) pkcs9(9) 16 }
id-aa     OBJECT IDENTIFIER ::= { id-smime 2 } -- attributes
id-aa-ets-archiveTimeStampV2 OBJECT IDENTIFIER ::= { id-aa 48 }
```

3.1.3　XAdES 仕様の調整

(1)　ETSI TR 102 038 v1.1.1 XML 署名ポリシの誤りの報告（2006 年 12 月 22 日）

XML 形式の署名ポリシフォーマットに関する技術レポートにおいて、何箇所かあった仕様の誤りを ETSI TC ESI の担当者に報告した。次版で反映されることとなった。

3.2　ISO TC171/SC2/WG5 への提案

ISO TC171/SC2/WG5 は PDF/A の標準化を行っている。ECOM は、同 WG5 の国内委員会の協力を得て、PDF/A への長期署名組み込みの提案を行ってきた。以下にその経緯と現時点の状況を報告する。

3.2.1　問題認識

PDF1.4[3]をベースとする PDF/A(PDF/A-1)は 2005 年 10 月に ISO 19005-1 として発行された。その後、次のバージョンとして、PDF1.6[4]をベースとする PDF/A(PDF/A-2)の標準化作業が始まった。しかしながら、昨年度の報告書の第 2 部第 1 章で述べたように、PDF1.6 は、仕様上の制約により長期署名の組み込みができないことが判明した（つまり、PDF/A-1 は長期署名の組み込みが可能であるが、PDF/A-2 は組み込みができず、このままでは後方互換性がなくなる恐れがあった)。

PDF においては、署名やその検証は署名ハンドラを呼びだす構造になっており、このときの署名ハンドラの名前や符号化方法などの名前を、Filter 名及び Subfilter 名という。長期署名処理を行う署名ハンドラを呼び出すには、それを識別するための Filter 名や Subfilter 名を付与する必要がある。名前なしで、必要な処理を呼び出すことはできない。

PDF1.4 仕様では、Filter 名、Subfilter 名に関して特に制約はなかった。一方、PDF1.6 仕様では次のように定義されている（表 3.2.1 参照)。

Filter 名：必須。Filter 名の例が示されている。

Subfilter 名：オプション。定義された Subfilter 名が列挙されている。長期署名のための名前は見当たらない。言うまでもなく、オプションだからという理由で勝手な名前を付けて構わないという訳ではない。

表 3.2.1.1: 署名辞書のエントリ定義

TABLE 8.98 Entries in a signature dictionary		
KEY	TYPE	VALUE
Type	name	(Optional) The type of PDF object that this dictionary describes; if present, must be Sig for a signature dictionary.
Filter	name	(Required; inheritable) The name of the preferred signature handler to use when validating this signature. If the Prop_Build entry is not present, it is also the name of the signature handler that was used to create the signature.（途中省略） An application may substitute a different handler when verifying the signature, as long as it supports the specified SubFilter format. Example signature handlers are Adobe.PPKLite, Entrust.PPKEF, CICI.SignIt, and VeriSign.PPKVS.
SubFilter	name	(Optional) A name that describes the encoding of the signature value and key information in the signature dictionary. An application may use any handler that supports this format to validate the signature. Defined values for public-key cryptographic signatures are adbe.x509.rsa_sha1, adbe.pkcs7.detached, and adbe.pkcs7.sha1 (see Section 8.7.2, “Signature Interoperability”).
Contents	string	(Required) The signature value. When ByteRange is present, the value is a hexadecimal string (see “Hexadecimal Strings” on page 32) representing the value of the byte range digest. （以下省略）

Subfilter 名はオプションであるが、PDF1.6 の 8.7.2 に書いてある通り異なる製品間の相互運用には不可欠である。しかし、長期署名に対応した名前は、 PDF1.6 には載っていない。

8.7.2 Signature Interoperability

It is intended that PDF consumer applications allow interoperability between signature handlers; that is, a PDF file signed with a handler from one vendor must be able to be validated with a handler from a different vendor.
The SubFilter entry in the signature dictionary specifies the encoding of the signature value and key information, and the Filter entry specifies the preferred handler to use to validate the signature. Handlers specify the SubFilter encodings they support; therefore, handlers other than the preferred handler can be used to validate the signature if necessary or desired.

図 3.2.1.1: PDF 仕様における署名の相互運用性の記述

3.2.2　提案の経緯と結論

従前より、長期署名の調査検討において、ECOM と社団法人日本画像マネジメント協会(JIIMA)は協力関係にあり、また、JIIMA が ISO TC171/SC2/WG5 の国内委員会の事務局を担当していることもあって、PDF/A の標準化状況を早くからウォッチしていた。PDF-A2 には前述のような問題があったため、2006 年 6 月 26 日～27 日に Costa Mesa で開催された WG5 委員会に、問題提起を行い、2007 年 1 月 22 日～23 日に開催された Orlando 会議で、長期署名を含む電子署名は PDF/A の規定とは独立した規定とすべきという日本の提案が各国の賛同を得て認められ、別の標準として（即ち、New Work Iterm (NWI)として）検討が行われることとなった。

なお、この会議で、PDF/A-2 がベースとする仕様は PDF1.6 から PDF1.7[5]に更新された。

付録１　暗号メッセージ構文を利用した電子署名(CAdES)の長期署名プロファイルに関する要求事項　JIS 案

本書は、今後の JIS 化審議過程において変更される可能性があります。

目　次

まえがき

この規格は，工業標準化法第 12 条第 1 項の規定に基づき，次世代電子商取引推進協議会(ECOM)から，工業標準原案を具して日本工業規格を制定すべきとの申出があり，日本工業標準調査会の審議を経て，経済産業大臣が制定した日本工業規格である。

この規格は，著作権法で保護対象となっている著作物である。

この規格の一部が，特許権，出願公開後の特許出願，実用新案権又は出願公開後の実用新案登録出願に抵触する可能性があることに注意を喚起する。経済産業大臣及び日本工業標準調査会は，このような特許権，出願公開後の特許出願，実用新案権又は出願公開後の実用新案登録出願に係る確認について，責任をもたない。

暗号メッセージ構文を利用した電子署名(CAdES)の長期署名プロファイルに関する要求事項

Long term signature profiles for cryptographic message syntax advanced electronic signatures(CAdES)

序文

この規格は，電子署名を長期にわたって検証可能にする長期署名に関する実装間の相互運用性を確保することを目的とする。

1　適用範囲

この規格は，長期署名プロファイルのうち，暗号メッセージ構文を利用した電子署名のプロファイルに関する要求事項について規定する。

2　引用規格

次に掲げる規格は，この規格に引用されることによって，この規格の規程の一部を構成する。これらの引用規格は，その最新版（追補を含む。）を適用する。

JIS X 0001　情報処理用語－基本用語

JIS X 0008　情報処理用語－セキュリティ

3　用語及び定義

この規格で用いる主な用語及び定義は，**JIS X 0001** 及び **JIS X 0008** によるほか，次による。

3.1

長期署名(long term signature)

署名時刻の特定並びに署名対象及び検証情報を含む署名に関する情報の改ざん(竄)検知を可能とする措置を実施し，署名を長期にわたって検証可能にした署名。

3.2

プロファイル(profile)

参照する仕様の選択要素や値の範囲に関する相互運用性要件。

3.3

要求レベル(required level)

プロファイルを構成する各要素の実装に対する要求の程度。

3.4

暗号メッセージ構文，CMS(cryptographic message syntax)

任意のメッセージ内容に対する，署名，ダイジェスト，認証及び暗号化に関する構文。

3.5

暗号メッセージ構文を利用した電子署名，CAdES(CMS advanced electronic signature)

署名者の識別やデータの改ざん(竄)検知が可能な暗号メッセージ構文に基づく電子署名。

3.6

暗号メッセージ構文を利用した時刻付き署名，CAdES-T(CAdES with time)

信頼のおける署名時刻（例えば，署名タイムスタンプ）を伴う暗号メッセージ構文に基づく電子署名。

3.7

暗号メッセージ構文を利用した長期保存署名，CAdES-A(archival CAdES)

署名対象及び検証情報を含む署名に関する情報の改ざん(竄)検知が可能な措置（例えば，アーカイブタイムスタンプ）を伴う暗号メッセージ構文に基づく電子署名。

3.8

コンテント情報(content information)

暗号メッセージ構文のコンテントを定義するデータ構造。

3.9

署名付きデータ(signed data)

暗号メッセージ構文における署名データを定義するデータ構造，またはそのデータ。

3.10

署名者情報(signerInfo)

署名者毎の署名情報を定義するデータ構造，またはそのデータ。

3.11

署名属性(signed attribute)

署名の対象となる署名情報。

3.12

非署名属性(unsigned attribute)

署名対象外の署名情報。

注記 署名タイムスタンプ及びアーカイブタイムスタンプなどは非署名属性である。

3.13

検証情報(validation data)

署名及びタイムスタンプの検証に用いる証明書及び失効情報。

3.14

タイムスタンピング機関，TSA(time-stamping authority)

あるデータが時間軸上のある点より前に存在していた証拠を提供することを信託された信頼できる第三者機関。

3.15

タイムスタンプトークン，TST(time-stamp token)

データの表現と時刻の値との間の，検証可能な暗号化された結合を含むデータ構造。タイムスタンプトークンは，その結合の中に追加のデータを含んでよい。

3.16

署名タイムスタンプ(signature timestamp)

署名が存在した時刻を特定可能にするために，署名値(SignatureValue)に付されるタイムスタンプ

3.17

アーカイブタイムスタンプ(archive timestamp)

改ざん(竄)を検知可能にするために，署名対象及び検証情報を含む署名に関する情報に付されるタイムスタンプ。

3.18

信頼点(trust anchor)

電子署名の検証を行う際の検証者によって信頼された信用の起点。公開かぎ(鍵)証明書または公開かぎ(鍵)の形式で提供され，一般的には，信頼する最上位の証明機関の公開かぎ(鍵)証明書である。

3.19

信頼できる第三者機関，TTP (trusted third party)

セキュリティ関連の活動に関して,他のエンティティから信頼されるセキュリティ機関又はその代理者。

3.20

証明機関，CA (certificate authority)

公開かぎ(鍵)証明書の作成及び割り当てを信託されたセンタ。任意選択として，証明機関は，エンティティに対してかぎ(鍵)を作成し，割り当てることもできる。

3.21

証明書(certificate)

あるエンティティの非対称かぎ(鍵)の一対のうち，公開されるかぎ(鍵)の情報で，証明機関によって署名され，それによって偽造できないようにしたもの。

3.22

属性証明書(attribute certificate)

職責，資格，地位などの属性及び属性値を記載した証明書。

3.23

失効情報(revocation information)

有効期間内に失効した証明書に関して証明機関が発行する情報。この情報と照合することで，証明書が現在も有効であるか否かを検証できる。

3.24

拡張セキュリティサービス，ESS (enhanced security service)

署名者証明書などの安全性を向上させるためのサービス。

4　規格適合性

この規格に適合する次の実装 **a)～d)**は，この規格が規定する CAdES-T プロファイル又は CAdES-A プロファイルにおける要求レベルが“必す(須)”の要素の処理をすべて備え，かつ，要求レベルが“要別途規定”の要素の処理を備えている場合は，その処理に関する詳細な仕様が定義されていなければならない。

a)　CAdES-T データの生成

b)　CAdES-T データの検証

c)　CAdES-A データの生成

d)　CAdES-A データの検証

規格適合性の宣言は，実装者自らが，実装状況（及び“要別途規定”要素に関してはその仕様）をプロファイル適合性宣言**(附属書 F)**に記載しこれを開示することによって行う。

注記 CAdES-T データの生成及び検証，並びに CAdES-A データの生成及び検証の位置付けについては**図 1** 参照。

5　長期署名プロファイル

5.1　定義する長期署名プロファイル

電子署名を長期にわたって検証可能にするためには，相互運用性が確保されていることのほかに，署名時刻の特定が可能であることに加え，署名対象及び検証情報を含む署名に関する情報の改ざん(竄)検出が可能であることが必要である。この規格では，CAdES に関して，次の二つのプロファイルを定義することによって，この要求を満たす。

a)　CAdES-T プロファイル　CAdES-T データの生成及び検証に関するプロファイル。

b)　CAdES-A プロファイル　CAdES-A データの生成及び検証に関するプロファイル。

ここで，CAdES-T データと CAdES-A データの関係を**図 1** に示す。

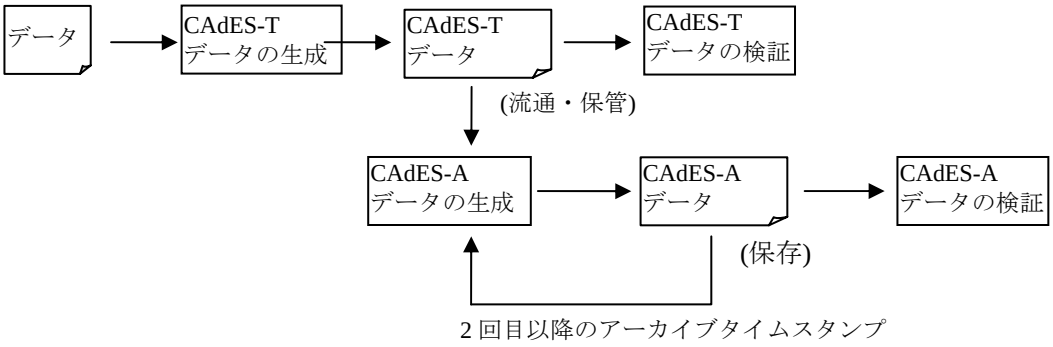


図 1－CAdES-T データと CAdES-A データとの関係

5.2　要求レベルの標語

この規格では，CAdES-T データ及び CAdES-A データを構成する各要素に対する，プロファイルとしての要求レベルとして，次の標語を定義する。

a)　必す(須)　この要求レベルをもつ要素は必ず実装しなければならない。この要求レベルの要素が，選択肢となる下位要素をもつ場合は，少なくともその一つを選択しなければならない。

b)　任意選択　この要求レベルをもつ要素の実装は任意である。

c)　要別途規定　この要求レベルをもつ要素の実装は任意であるが，その処理に関して，別途に詳細な仕様を規定しなければならない。

5.3　要求レベルの設定基準

CAdES-T 又は CAdES-A データを構成する各要素に対する要求レベルの設定は，次の基準に基づく。

a)　CAdES の定義で“必す(須)”となっている要素，並びに長期署名の生成及び検証に最低限必要な要素は“必す(須)”とする。

b)　外部定義要素は，“要別途規定”とする。

　　例　　その他形式の証明書

c)　特定アプリケーションとの連携を前提とした要素は，“要別途規定”とする。

　　例　　コンテント参照情報

d)　運用に依存する要因が残っている要素は，“要別途規定”とする。

　　例　　属性証明書系要素，タイムマークなど

注記　ISO/IEC 1804-2 で定義されたアーカイビング方式のタイムスタンプはその他タイムスタンプに含まれる。

e) 単なる参考情報を運ぶ要素に関しては，その要素を実装していない場合，要求を無視する。

例　署名時刻

5.4　CAdES-T プロファイル

5.4.1 コンテント情報

CAdES-T を構成するコンテント情報は**表 1** に示す要求レベルに従う。

表 1－コンテント情報

要素	要求レベル	条件/値	参照箇条
コンテント種別	必す(須)	署名付きデータを表す識別子	**A.2.1**
コンテント	必す(須)	署名付きデータ	**A.2.2**

5.4.2　署名付きデータ及び署名者情報

署名付きデータ及び署名者情報の各要素は，**表 2** 及び**表 3** に示す要求レベルに従う。

表 2－署名付きデータ

要素	要求レベル	条件/値	参照箇条
暗号メッセージ構文の版数	必す(須)		**A.3.1**
ダイジェストアルゴリズム識別子群	必す(須)		**A.3.2**
カプセル構造化されたコンテント情報	必す(須)		**A.3.3**
・・e コンテント種別（オブジェクト識別子）	必す(須)		**A.3.3.1**
・・e コンテント	任意選択		**A.3.3.2**
証明書群	任意選択		**A.3.4**
・・証明書	任意選択		**A.3.4.**
・・属性証明書 2 版	要別途規定		**A.3.4.**
・・その他形式の証明書	要別途規定		**A.3.4.**
失効情報群	任意選択		**A.3.5**
・・失効情報	任意選択		**A.3.5.1**
・・その他形式の失効情報	要別途規定		**A.3.5.2**
署名者情報群	必す(須)		**A.3.6**

表 3－署名者情報

要素	要求レベル	条件/値	参照箇条
暗号メッセージ構文の版数	必す(須)		**A.4.1**
署名者識別子	必す(須)		**A.4.2**
・・発行者及びシリアル番号	任意選択		**A.4.2.1**
・・対象者かぎ(鍵)識別子	任意選択		**A.4.2.2**
ダイジェストアルゴリズム識別子	必す(須)		**A.4.3**
署名属性群	必す(須)		**A.4.4**
署名アルゴリズム識別子	必す(須)		**A.4.5**
署名値	必す(須)		**A.4.6**
非署名属性群	必す(須)		**A.4.7**

5.4.3 署名属性及び非署名属性

CAdES で定義された署名属性群及び非署名属性群の各要素は，**表 4** 及び**表 5** に示す要求レベルに従う。これらの表に定義されていない要素の要求レベル“要別途規定”である。

表 4－署名属性

要素	要求レベル	条件/値	参照箇条
コンテント種別	必す(須)		**A.5.1**
メッセージダイジェスト	必す(須)		**A.5.2**
署名者証明書の参照情報	必す(須)		**A.5.3**
・・ESS 署名者証明書の参照情報	任意選択 a)		**A.5.3**
・・ESS 署名者証明書の参照情報 2 版	任意選択 a)		**A.5.3**
・・他の署名者証明書の参照情報	要別途規定 a)		**A.5.3**
署名ポリシ識別子	任意選択		**A.5.4**
署名時刻	任意選択 b)		**A.5.5**
コンテント参照情報	要別途規定		**A.5.6**
コンテント識別子	要別途規定		**A.5.7**
コンテントのヒント	要別途規定		**A5.8**
コミットメント識別表示	要別途規定		**A5.9**
署名者所在地	要別途規定		**A.5.10**
署名者の属性情報	要別途規定		**A.5.11**
コンテントタイムスタンプ	要別途規定		**A.5.12**
注 a) ESS 署名者証明書の参照情報，ESS 署名者証明書の参照情報 2 版，他の署名者証明書の参照情報のいずれか一つを選択。 b) 未実装の場合は無視。			

表 5－非署名属性

要素	要求レベル	条件/値	参照箇条
カウンタ署名	任意選択		**A.6.1**
署名タイムスタンプ	任意選択 a)		**A.6.2**
タイムマークなどその他の方式	要別途規定 a)		**A.6.3**
注 a) 署名タイムスタンプかタイムマークなどその他の方式のいずれか一つを選択。			

5.5 CAdES-A プロファイル

5.5.1 前提条件

CAdES-A プロファイルは，CAdES-T データの拡張として定義される。基となる CAdES-T データは，必ずしも **5.4** の CAdES-T プロファイルに準じる必要はない。

5.5.2 追加非署名属性

CAdES-T データの非署名属性群に追加する，CAdES で定義された各要素は，**表 6** に示す要求レベルに従う。この表に定義されていない要素の要求レベルは“要別途規定”である。

表 6－追加非署名属性

要素	要求レベル	条件/値	参照箇条
完全な証明書参照情報群	必す(須)		**A.6.4**
完全な失効参照情報群	必す(須)		**A.6.5**
・・CRL 形式の失効参照情報群	任意選択		**A.6.5**
・・OCSP 形式の失効参照情報群	任意選択		**A.6.5**
・・他の形式の失効参照情報群	要別途規定		**A.6.5**
属性証明書の参照情報群	要別途規定		**A.6.6**
属性失効情報の参照情報群	要別途規定		**A.6.7**
証明書群	必す(須)		**A.6.8**
・・証明書	任意選択		**A.6.8**
・・CA 等による証明書の保管	要別途規定		**A.6.8**
失効情報群	必す(須)		**A.6.9**
・・CRL による失効情報	任意選択		**A.6.9**
・・基本 OCSP 応答	任意選択		**A.6.9**
・・他の失効情報	要別途規定		**A.6.9**
・・CA 等による失効情報の保管	要別途規定		**A.6.9**
CAdES-C データへのタイムスタンプ	要別途規定		**A.6.10**
タイムスタンプが付与された証明書及び失効情報に関する参照	要別途規定		**A.6.11**
アーカイブタイムスタンプ id-aa-48	任意選択 a)		**A.6.12**
アーカイブタイムスタンプ id-aa-27	任意選択 a)		**A.6.12**
注 a) アーカイブタイムスタンプ id-aa-48 又はアーカイブタイムスタンプ id-aa-27 のどちらか又は両方が存在しなければならない。			

5.6 タイムスタンプの検証情報

過去のタイムスタンプを検証する際には，信頼点までの証明書及び失効情報が必要となる。タイムスタンプの検証情報とは，TSA の証明書から信頼点の証明書までの証明書チェーン上の証明書と、それぞれの失効情報である。

検証情報は、次に示す方法によって CAdES-A データ内に格納することができる。CAdES-A データ内に格納しない場合は、他の安全な手段によって保存されている必要がある。他の安全な手段の例は，TTP である CA 又は TSA が保存しておく方法である。

また、過去のタイムスタンプの検証の際には、次に示す方法によって格納された検証情報を利用してもよいし、他の安全な手段によって保存された検証情報を利用してもよい。

タイムスタンプトークンの構造に関しては，要求事項を**附属書 E** に示す。

a) 署名タイムスタンプの検証情報 生成時, 次のいずれかの場所に格納するか, 又は CA 等で保存する。

1) 非署名属性内の次の要素。

－証明書群

－失効情報群

2) 署名タイムスタンプにおけるタイムスタンプトークン（署名タイムスタンプトークン）内の次の要素。

－証明書群

－失効情報群

3) 署名タイムスタンプトークンの非署名属性群内の次の要素。

－証明書群

－失効情報群

注記　2)は，CMS で定義され，**3)**は，CAdES で定義されている。

b)　アーカイブタイムスタンプの検証情報　生成時，次のいずれかの場所に格納するか，又は CA 等で保存する。

1) アーカイブタイムスタンプにおけるタイムスタンプトークン（アーカイブタイムスタンプトークン）内の次の要素。

－証明書群

－失効情報群

2) アーカイブタイムスタンプトークンの非署名属性群内の次の要素。

－証明書群

－失効情報群

注記　1)は，CMS で定義され，**2)**は，CAdES で定義されている。

附属書 A
（規定）
CAdES のデータ構造と構成要素

序文

この附属書は，CAdES のデータ構造と構成要素を規定する。

A.1　CAdES のデータ構造

CAdES のデータ構造のベースは，署名付きデータである。署名付きデータは，署名対象をカプセル構造化して取り込み，複数の署名者の署名を格納することができる（**図 A.1** の破線部分）。**図 A.2 に**，署名付きデータの構造を示す。

注記　署名付きデータは，コンテント情報のテンプレートに従って，コンテント種別が“署名付きデータを表す識別子(id-signedData)”のコンテントとして定義される。

署名者毎の署名情報は，署名者情報として構造化されている（**図 A.3**）。署名者情報は，**表 A.1** に示す属性をもつことができる。これらの属性は，属性のテンプレートに従って定義されている。

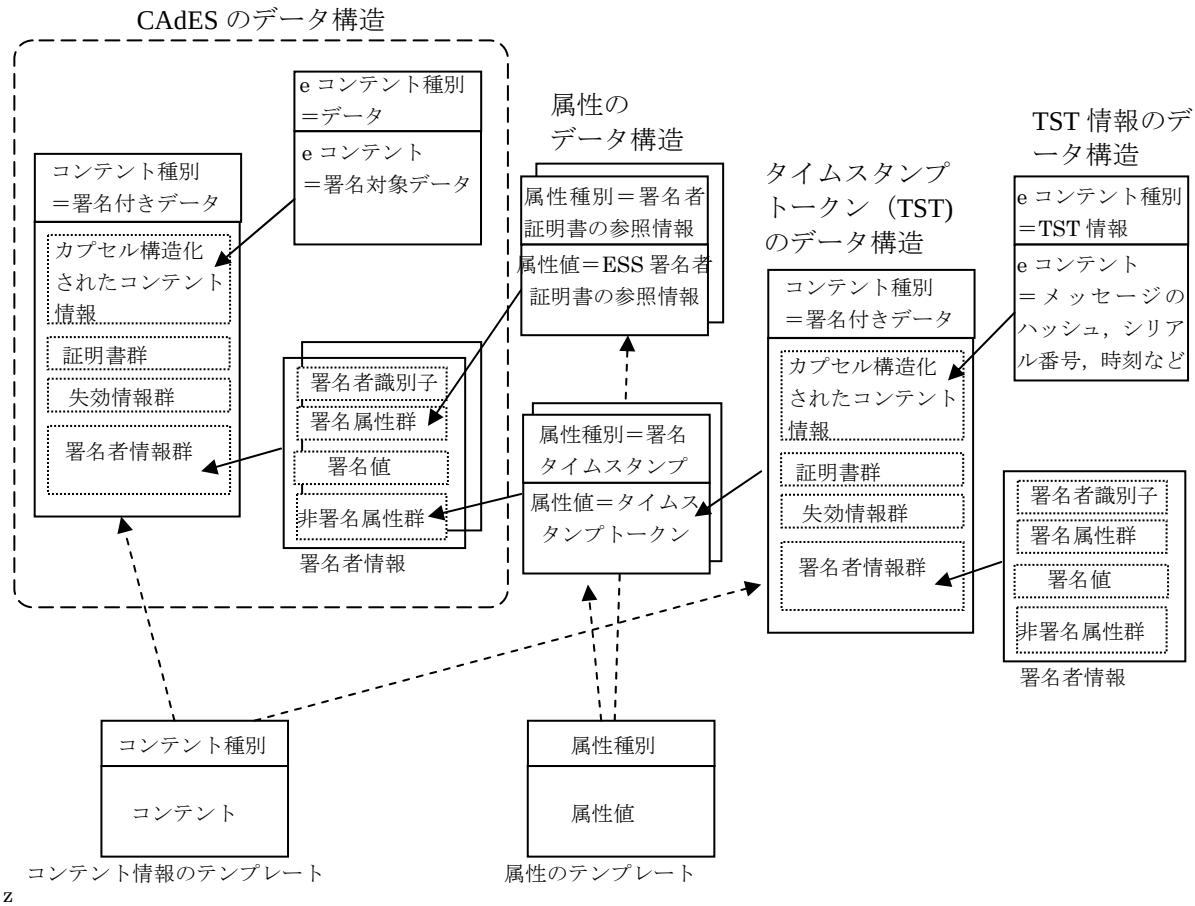


図 A.1－CAdES のデータ構造

<table>
<tr><td colspan="3">コンテント種別</td></tr>
<tr><td rowspan="5">コンテント</td><td colspan="2">暗号メッセージ構文の版数</td></tr>
<tr><td colspan="2">ダイジェストアルゴリズム識別子群</td></tr>
<tr><td rowspan="2">カプセル構造化された
コンテント情報</td><td>コンテント種別</td></tr>
<tr><td>コンテント</td></tr>
<tr><td colspan="2">証明書群</td></tr>
<tr><td colspan="3">失効情報群</td></tr>
<tr><td colspan="3">署名者情報群</td></tr>
</table>

図 A.2－署名付きデータの構造

署名者情報の版数
署名者識別子 （発行者及びシリアル番号　または　対象者かぎ（鍵）識別子）
ダイジェストアルゴリズム識別子
署名属性群
署名アルゴリズム識別子
署名値
非署名属性群

図 A.3－署名者情報の構造

表 A.1－属性一覧

分類	属性
署名属性	コンテント種別
	メッセージダイジェスト
	署名者証明書の参照情報
	署名ポリシ識別子
	署名時刻
	コンテント参照情報
	コンテント識別子
	コンテントのヒント
	コミットメント識別表示
	署名者所在地
	署名者の属性情報
	コンテントタイムスタンプ
非署名属性	カウンタ署名
	署名タイムスタンプ
	完全な証明書参照情報群
	完全な失効参照情報群
	属性証明書の参照情報群
	属性失効情報の参照情報群
	証明書群
	失効情報群
	CAdES-C データへのタイムスタンプ
	タイムスタンプが付与された証明書及び失効情報に関する参照情報
	アーカイブタイムスタンプ

A.2　コンテント情報の要素

A.2.1　コンテント種別

暗号メッセージの種別。これはオブジェクト識別子であり，この暗号メッセージの種別を定義した機関によって割り当てられた一意に定まる整数列である。

A.2.2　コンテント

暗号メッセージの内容。この内容は，コンテント種別によって一意に定められる。

A.3　署名付きデータの要素

A.3.1　暗号メッセージ構文の版数

署名付きデータの構文の版数。

A.3.2　ダイジェストアルゴリズム識別子群

ダイジェストアルゴリズムの集まり。

A.3.3　カプセル構造化されたコンテント情報

署名対象文書（データ）及びそれに関する情報。

A.3.3.1　e コンテント種別

署名対象文書（データ）のデータ型を一意に識別するオブジェクト識別子。

A.3.3.2　e コンテント

署名対象文書（データ）のデータバイト列を格納する。省略することによって，“外部署名”を構成することが可能になる。

A.3.4　証明書群

証明書の集まり。認識される“ルート”又は“最上位の証明機関”から署名者情報群に含まれるすべての署名者までの連鎖を含むことができる。

a)　証明書

b)　属性証明書 2 版

c)　その他形式の証明書

A.3.5　失効情報群

証明書失効リスト(CRL)の集まり。証明書群に含まれる証明書が有効か否かを決定するための情報を含むことができる。

a)　失効情報

b)　その他形式の失効情報

A.3.6　署名者情報群

署名者情報の集まり。0 個も含めて，どのような数の署名者情報も含むことがある。

A.4　署名者情報の要素

A.4.1　暗号メッセージ構文の版数

署名者情報の構文の版数である。署名者識別子が発行者及びシリアル番号ならば，値は 1 でなければならない。対象者かぎ(鍵)識別子ならば，値は 3 でなければならない。

A.4.2　署名者識別子

署名者の証明書を特定する（それによって署名者の公開かぎ(鍵)も特定する。）。署名者の公開かぎは，

受信者が署名を検証するのに必要である。署名者の公開かぎを特定するための二つの選択肢を提供する。

A.4.2.1　発行者及びシリアル番号

署名者の証明書を，発行者の識別名及び証明書のシリアル番号によって識別する。

A.4.2.2　対象者かぎ（鍵）識別子

署名者の証明書を X.509 の対象者かぎ（鍵）識別子の拡張値によって識別する。

A.4.3　ダイジェストアルゴリズム識別子

署名者に使われたメッセージダイジェストのアルゴリズム及び関連するパラメタを識別する。メッセージダイジェストのアルゴリズムは，関連する署名付きデータのダイジェストアルゴリズム識別子群領域に挙げられているものでなければならない。

A.4.4　署名属性群

署名される属性の集まり。任意選択であるが，存在する場合は，DER 符号化と，少なくとも，コンテント種別とメッセージダイジェストの二つの属性を含まなければならない。

A.4.5　署名アルゴリズム識別子

署名者に使われたメッセージダイジェストのアルゴリズム及び関連するパラメタを識別する。

A.4.6　署名値

署名の値。

A.4.7　非署名属性群

署名対象とはならない属性の集まり。

A.5　署名属性として定義される要素

A.5.1　コンテント種別

署名対象データのデータ型のオブジェクト識別子を保持する。署名属性が存在する場合には必ず含めなければならない。

A.5.2　メッセージダイジェスト

署名対象データのハッシュ値を保持する。署名属性が存在する場合には必ず含めなければならない。

A.5.3　署名者証明書の参照情報

署名者証明書を特定する情報（署名者証明書のハッシュ値）を保持する。

注記：本属性は署名者証明書を特定するという目的においては対象者かぎ（鍵）識別子と同じであるが、対象者かぎ（鍵）識別子は署名保護されるフィールドではないため、改ざん(竄)されてもその事実を署名検証時に検知することができないという脆弱性を補うことができる。

a)　ESS 署名者証明書の参照情報

b)　ESS 署名他の署名者証明書の参照情報

A.5.4　署名ポリシ識別子

署名ポリシを特定するための情報を保持する。署名ポリシは、署名者と署名の検証者が守るべき一連の規則。

A.5.5　署名時刻

署名がなされた時刻を保持する。時刻の正確さは要求されない。

A.5.6　コンテント参照情報

他の署名文書へのリンクを保持する。

A.5.7　コンテント識別子

署名対象文書のハッシュ値など，コンテントを一意に特定する情報を保持する。

A.5.8　コンテントのヒント

署名対象文書のデータフォーマットに関する補足情報を保持する。

A.5.9　コミットメント識別表示

署名者がどのような意図を持って署名したかを表明する情報を保持する。

A.5.10　署名者所在地

署名者の署名時における居所情報を格納する属性である。第三者によってその確かさが保証されたものではなく、署名者が署名時にその場所にいたと主張しているものである。

A.5.11　署名者の属性情報

任意の署名者属性情報を保持する。署名者属性情報には大きく分けて，署名者が自分でそうだと主張している属性情報と，署名者がそうであると第三者が保証している属性情報の２種類があり、本属性にはそのどちらの属性情報も格納することができる。

A.5.12　コンテントタイムスタンプ

署名対象文書に対するタイムスタンプ。署名時点より前に取っておいた署名対象文書の存在証明（つまりタイムスタンプ）を署名データに含めたい場合に利用する。

A.6　非署名属性として定義される要素

A.6.1　カウンタ署名

署名値に対する署名。複数人が同一文書に署名する際、署名の順序に意味がある、もしくは意味を持たせたい場合に用いる。

A.6.2　署名タイムスタンプ

署名が存在した時刻を特定可能にするために，署名値に付されるタイムスタンプ。

A.6.3　タイムマークなどその他の方式

データが特定の時刻以前に存在したことを示すための，データと特定の時間を結びつける信頼される第三者機関からの監査証跡内の情報。

A.6.4　完全な証明書参照情報群

署名検証に必要な，署名者証明書から信頼点の証明書までの認証パス上の全ての証明書の参照情報（ただし署名者の証明書への参照情報は含まない)。１署名につき一つだけ存在する。

A.6.5　完全な失効参照情報群

署名検証に必要な, 署名者から信頼点までの証明書に対する CRL あるいは OCSP 応答のすべての参照情報。１署名に対して一つだけ存在する。

a)　CRL 形式の失効参照情報群

b)　OCSP 形式の失効参照情報群

c)　他の形式の失効参照情報群

A.6.6　属性証明書の参照情報群

関連する属性証明書の参照情報を保持する。

A.6.7　属性失効情報の参照情報群

関連する属性証明書の失効情報（ACRL または OCSP レスポンス)の参照情報を保持する。

A.6.8　証明書群

完全な証明書参照情報群が参照する証明書および署名者の証明書を保持する。1 署名につき一つだけ存

在する。

a) 証明書群

A.6.9 失効情報群

完全な失効参照情報群で参照される CRL と OCSP 応答の値を保持する。1 署名につき一つだけ存在する。

a) CRL による失効情報

b) 基本 OCSP 応答

c) 他の失効情報

A.6.10 CAdES-C データへのタイムスタンプ

認証経路を構成する全証明書参照情報付き署名(CAdES-C)全体に対するタイムスタンプ。1 署名につき複数存在可能。

A.6.11 タイムスタンプが付与された証明書及び失効情報に関する参照情報

検証情報へのリファレンス（完全な証明書参照情報群および完全な失効参照情報群）に対するタイムスタンプ。

A.6.12 アーカイブタイムスタンプ

改ざん(竄)を検知可能にするために，署名対象及び検証情報を含む署名に関する情報に付されるタイムスタンプ。

注記 id-aa-48 及び id-aa-27 はそれぞれ，ETSI TS 101 733 V1.7.3 及び ETSI TS 101 733 V1.2.2 で定義されている。

附属書 B
（参考）
参照文献

序文

この附属書は，長期署名プロファイルに関する要求事項に沿った実装が参照する仕様について記載するものであって，規定の一部ではない。

B.1　CAdES 仕様

ETSI TS 101 733 CMS Advanced Electronic Signatures (CAdES)

注記　http://pda.etsi.org/pda/queryform.asp から入手可能。

B.2　CMS 仕様

IETF RFC 3852 Cryptographic Message Syntax

注記　http://www.ietf.org/rfc.html から入手可能。

附属書 C
（参考）
要素名と ASN.1 表記の対応表

序文

この附属書は，この規格で用いられる要素の ASN.1 表記について記載するものであって，規定の一部ではない。

表 C.1－コンテント情報(ContentInfo)

要素名	ASN.1 表記
コンテント種別	ContentType
コンテント	Content

表 C.2－署名付きデータ(SignedData)

要素名	ASN.1 表記
暗号メッセージ構文の版数	CMSVersion
ダイジェストアルゴリズム識別子群	DigestAlgorithmIdentifiers
カプセル構造化されたコンテント情報	EncapsulatedContentInfo
・・e コンテント種別	・・eContentType
・・e コンテント	・・eContent
証明書群	CertificateSet　(Certificates)
・・証明書	・・Certificate
・・属性証明書 2 版	・・AttributeCertificateV2
・・その他形式の証明書	・・OtherCertificateFormat
失効情報群	RevocationInfoChoices　(crls)
・・失効情報	・・CertificateList
・・その他形式の失効情報	・・OtherRevocationInfoFormat
署名者情報群	SignerInfos

表 C.3－署名者情報(SignerInfo)

要素名	ASN.1 表記
暗号メッセージ構文の版数	CMSVersion
署名者識別子	SignerIdentifier
・・発行者及びシリアル番号	・・IssuerAndSerialNumber
・・対象者かぎ（鍵）識別子	・・SubjectKeyIdentifier
ダイジェストアルゴリズム識別子	DigestAlgorithmIdentifier
署名属性群	SignedAttributes
署名アルゴリズム識別子	SignatureAlgorithmIdentifier
署名値	SignatureValue
非署名属性群	UnsignedAttributes

表 C.4－署名属性(Signed Attribute)

要素名	ASN.1 表記
コンテント種別	ContentType
メッセージダイジェスト	MessageDigest
署名者証明書の参照情報	SigningCertificateReference
・・ESS 署名者証明書の参照情報	・・ESS SigningCertificate
・・ESS 署名者証明書の参照情報 2 版	・・ESS SigningCertificate v2
・・他の署名者証明書の参照情報	・・OtherSigningCertificate
署名ポリシ識別子	SignaturePolicyIdentifier
署名時刻(	SigningTime
コンテント参照情報	ContentReference
コンテント識別子	ContentIdentifier
コンテントのヒント	ContentHints
コミットメント識別表示	CommitmentTypeIndication
署名者所在地	SignerLocation
署名者の属性情報	SignerAttribute
コンテントタイムスタンプ	ContentTomestamp

表 C.5－非署名属性(Unsigned Attribute)

要素名	ASN.1 表記
カウンタ署名	CounterSignature
署名タイムスタンプ	SignatureTimeStamp
タイムマークなどその他の方式	

表 C.6－追加非署名属性(Unsigned Attribute)

要素名	ASN.1 表記
完全な証明書参照情報群	CompleteCertificateReferences
完全な失効参照情報群	CompleteRevocationReferences
・・CRL 形式の失効参照情報群	・・CompleteRevRefs CRL
・・OCSP 形式の失効参照情報群	・・CompleteRevRefs OCSP
・・他の形式の失効参照情報群	・・OtherRevRefs
属性証明書の参照情報群	Attribute certificate references
属性失効情報の参照情報群	Attribute revocation references
証明書群	CertificateValues
・・証明書	・・CertificateValues
・・CA 等による証明書の保管	・・
失効情報群	RevocationValues
・・CRL による失効情報	・・CertificateList
・・基本 OCSP 応答	・・BasicOCSPResponse
・・他の失効情報	・・OtherRevVals
・・CA 等による失効情報の保管	・・
CAdES-C データへのタイムスタンプ	CAdES-C-timestamp
タイムスタンプが付与された証明書及び失効情報に関する参照	Time-stamped cert and crls reference
アーカイブタイムスタンプ	ArchiveTimestamp

附属書 D
（規定）
タイムスタンプトークンの構造

序文

この附属書は，長期署名におけるタイムスタンプトークンの構造に関する要求事項を規定する。

D.1　準拠する仕様

この規格における署名タイムスタンプトークン及びアーカイブタイムスタンプトークンは次の仕様に準拠する。

a)　基本構造　CMS に準拠する。

b)　署名属性及び非署名属性　CAdES に準拠する。

D.2　構成要素の要求レベル

署名タイムスタンプトークン及びアーカイブタイムスタンプトークンの各要素に対する要求レベルは**表 D.1** に従う。

表 D.1－タイムスタンプトークンの各要素に対する要求レベル

要素	要求レベル	条件又は値
コンテント種別	必す(須)	署名付きデータを表す識別子
コンテント	必す(須)	署名付きデータ
・・暗号メッセージ構文の版数	必す(須)	
・・ダイジェストアルゴリズム識別子群	必す(須)	
・・カプセル構造化されたコンテント情報	必す(須)	
・・・・e コンテント種別	必す(須)	TST 情報を表す識別子
・・・・e コンテント	必す(須)	TST 情報
・・証明書群	任意選択	
・・・・証明書	任意選択	
・・・・属性証明書 1 版	任意選択	
・・・・属性証明書 2 版	任意選択	
・・・・その他形式の証明書	要別途規定	
・・失効情報群	任意選択	
・・・・失効情報	任意選択	
・・・・その他形式の失効情報	任意選択	
・・署名者情報群	必す(須)	

表 D.1－タイムスタンプトークンの各要素に対する要求レベル（続き）

要素	要求レベル	条件又は値
・・・・署名者情報	必す(須)	
・・・・・・暗号メッセージ構文の版数	必す(須)	
・・・・・・署名者識別子	必す(須)	
・・・・・・・・発行者及びシリアル番号	任意選択	
・・・・・・・・対象者かぎ（鍵）識別子	任意選択	
・・・・・・ダイジェストアルゴリズム識別子	必す(須)	
・・・・・・署名属性群	必す(須)	
・・・・・・・・コンテント種別	必す(須)	TST 情報を表す識別子
・・・・・・・・メッセージダイジェスト	必す(須)	
・・・・・・・・署名者証明書の参照情報	必す(須)	
・・・・・・・・・・ESS 署名者証明書の参照情報	任意選択 a)	
・・・・・・・・・・ESS 署名者証明書の参照情報 2 版	任意選択 a)	
・・・・・・・・・・他の署名者証明書の参照情報	任意選択 a)	
・・・・・・署名アルゴリズム識別子	必す(須)	
・・・・・・署名値	必す(須)	
・・・・・・非署名属性群	任意選択	
・・・・・・・・完全な証明書参照情報群	任意選択	
・・・・・・・・完全な失効参照情報群	任意選択	
・・・・・・・・・・CRL 形式の失効参照情報群	任意選択	
・・・・・・・・・・OCSP 形式の失効参照情報群	任意選択	
・・・・・・・・証明書群	任意選択	
・・・・・・・・・・証明書	任意選択	
・・・・・・・・・・CA 等による証明書の保管	要別途規定	
・・・・・・・・失効情報群	任意選択	
・・・・・・・・・・CRL による失効情報	任意選択	
・・・・・・・・・・基本 OCSP 応答	任意選択	
・・・・・・・・・・他の失効情報	要別途規定	
注 a) ESS 署名者証明書の参照情報，ESS 署名者証明書の参照情報 2 版，他の署名者証明書の参照情報のいずれか一つを選択。		

附属書 E
（参考）
PDF/A への長期署名の適用方法

序文

ISO 19005-1 として標準化された、長期保存を目的とした PDF/A は、電子署名を付与することが可能であるが、現時点においては，署名ハンドラ及び符号化方法に対する名前付け規則が決まっておらず，また，アーカイブタイムタイムスタンプを用いた長期署名において、電子署名データのサイズ増加に伴う署名対象範囲の扱い方，及び可変長署名データの格納方法が標準化されておらず，実装依存となることにより相互運用性が確保できないことが懸念される。

このため、この附属書では，相互運用性確保に必要な名前付け規則，署名対象範囲の指定方法及び署名データの格納方法について参考のために記載するものであって，規定の一部ではない。

E.1　フィルタ名及びサブフィルタ名の名前付け規則

a)　名前の基本構造は，次の通りとする。

“商標登録されているなど一意性のある製造者名”＋“ドット”＋“製造者名のもとで一意な名前”

例　ecom . example

b)　この附属書では，PDF/A への長期署名適用のために，次の名前を定義する。

1)　**.AdES**　長期署名のための署名ハンドラの名前（フィルタ名）

2)　**jis.cades**　CAdES 処理を示す名前（サブフィルタ名）

3)　**jis.xades**　XAdES 処理を示す名前（サブフィルタ名）

D.2　バイトレンジの指定方法

署名範囲を示すバイトレンジは**表 E.1** に従う。

注記　**図 E.1** のように，時刻付き署名データ部分を避けて①及び②の範囲を指定する。

表 E.1 - バイトレンジ

配列中の位置	値	注記
0	ファイルの先頭	ゼロ
1	配列 0 の位置から署名タグ内の Contents の実データ直前までのバイト長さ	“<”までの長さ
2	署名タグ内の Contents の実データ直後のバイトオフセット	“>”の位置
3	配列 2 の位置から長期保存署名データ（添付ファイル）の直前までのバイト長さ	

E.3　時刻付き署名データ及び長期保存署名データの格納方法

時刻付き署名データを PDF コンテンツの署名タグ内に格納し，可変長の長期保存署名データは添付ファイルとして格納する（**図 E.1** 参照）。

a) 署名タグ内に“/sigAdES”を記述し，長期保存署名データを格納した添付ファイルの存在を表示する。

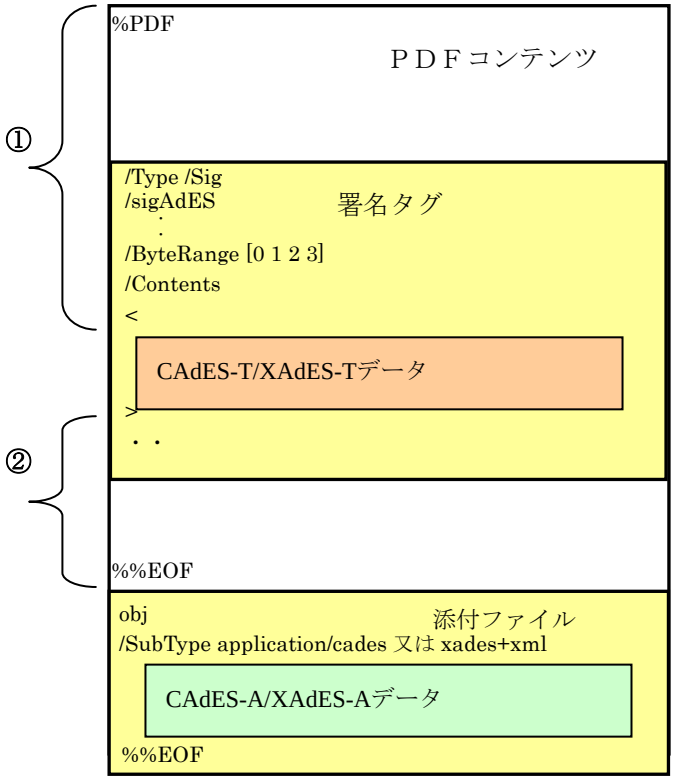
b) 添付ファイルの MIME タイプは“application/cades”又は“application/xades+xml”とする。

c) 署名タグ内の時刻付き署名データと，添付ファイルに格納する長期保存署名データ内の時刻付き署名データ部分は同一とする。

注記　署名タグ内に時刻付き署名データを格納するタイミングで，同時に，添付ファイル内にも同じ時刻付き署名データを格納する。

d) 添付ファイルに格納する長期保存署名データはバイナリ変換せず、そのまま格納する。

e) 長期署名の延長（アーカイブタイムスタンプの再取得）は，添付ファイルに格納された長期保存署名データに対して行い，添付ファイルを差し替える。



注記　図では，相互参照セクションやトレーラなどは省略している。

図 E.1 - PDF ファイルへのデータ格納方法

附属書 F
（規定）
プロファイル適合性宣言

序文

この附属書は，CAdES の実装に対するプロファイルの自己適合宣言のための書式を規定する。

記入方法

実装されている要素を“生成”及び“検証”欄にチェックマーク（✔）で示す。

F.1　記入者及び記入日

記入者	
記入日	

F2　実装の名称

名称	
版番号	

F.3　参照する CAdES のバージョン

バージョン番号	

E.4　プロファイル適合性を宣言する実装の範囲

表 F.1－実装範囲

	生成	検証
CAdES-T		
CAdES-A		

F.5　CAdES-T プロファイルへの適合

表 F.2－署名付きデータ

要素	要求レベル	条件/値	参照箇条	生成	検証
暗号メッセージ構文の版数	必す(須)		**A.3.1**		
ダイジェストアルゴリズム識別子群	必す(須)		**A.3.2**		
カプセル構造化されたコンテント情報	必す(須)		**A.3.3**		
・・コンテント種別	必す(須)		**A.3.3.1**		
・・コンテント	任意選択		**A.3.3.2**		
証明書群	任意選択		**A.3.4**		
・・証明書	任意選択		**A.3.4.1**		
・・属性証明書 2 版	要別途規定		**A.3.4.2**		
・・その他形式の証明書	要別途規定		**A.3.4.3**		

表 F.2－署名付きデータ（続き）

要素	要求レベル	条件/値	参照箇条	生成	検証
失効情報の群	任意選択		**A.3.5**		
・・失効情報	任意選択		**A.3.5.1**		
・・その他形式の失効情報	要別途規定		**A.3.5.2**		
署名者情報群	必す(須)		**A.3.6**		

表 F.3－署名者情報

要素	要求レベル	条件/値	参照箇条	生成	検証
暗号メッセージ構文の版数	必す(須)		**A.4.1**		
署名者識別子	必す(須)		**A.4.2**		
・・発行者及びシリアル番号	任意選択		**A.4.2.1**		
・・対象者かぎ（鍵）識別子	任意選択		**A.4.2.2**		
ダイジェストアルゴリズム識別子	必す(須)		**A.4.3**		
署名属性群	必す(須)		**A.4.4**		
署名アルゴリズム識別子	必す(須)		**A.4.5**		
署名値	必す(須)		**A.4.6**		
非署名属性群	必す(須)		**A.4.7**		

表 F.4－署名属性

要素	要求レベル	条件/値	参照箇条	生成	検証
コンテント種別	必す(須)		**A.5.1**		
メッセージダイジェスト	必す(須)		**A.5.2**		
署名者証明書の参照情報	必す(須)		**A.5.3**		
・・ESS 署名者証明書の参照情報	任意選択 a)		**A.5.3**		
・・ESS 署名者証明書の参照情報 2 版	任意選択 a)		**A.5.3**		
・・他の署名者証明書の参照情報	任意選択 a)		**A.5.3**		
署名ポリシ識別子	任意選択		**A.5.4**		
署名時刻	任意選択 b)		**A.5.5**		
コンテント参照情報	要別途規定		**A.5.6**		
コンテント識別子	要別途規定		**A.5.7**		
コンテントのヒント	要別途規定		**A5.8**		
コミットメント識別表示	要別途規定		**A5.9**		
署名者所在地	任意選択		**A.5.10**		
署名者の属性情報	任意選択		**A.5.11**		
コンテントタイムスタンプ	要別途規定		**A.5.12**		
注 a) ESS 署名者証明書の参照情報，ESS 署名者証明書の参照情報 2 版，他の署名者証明書の参照情報のいずれか一つを選択。 b) 未実装の場合は無視。					

表 F.5－非署名属性

要素	要求レベル	条件/値	参照箇条	生成	検証
カウンタ署名	任意選択		**A.6.1**		
署名タイムスタンプ	任意選択 a)		**A.6.2**		
タイムマークなどその他の方式	要別途規定 a)		**A.6.3**		
注 a) 署名タイムスタンプかタイムマークなどその他の方式のいずれか一つを選択					

F.6　CAdES-A プロファイルへの適合

表 F.6－追加非署名属性

要素	要求レベル	条件/値	参照箇条	生成	検証
完全な証明書参照情報群	必す(須)		**A.6.4**		
完全な失効参照情報群	必す(須)		**A.6.5**		
・・CRL 形式の失効参照情報群	任意選択		**A.6.5**		
・・OCSP 形式の失効参照情報群	任意選択		**A.6.5**		
・・他の形式の失効参照情報群	要別途規定		**A.6.5**		
属性証明書の参照情報群	要別途規定		**A.6.6**		
属性失効情報の参照情報群	要別途規定		**A.6.7**		
証明書群	必す(須)		**A.6.8**		
・・証明書	任意選択		**A.6.8**		
・・CA 等による証明書の保管	要別途規定		**A.6.8**		
失効情報群	必す(須)		**A.6.9**		
・・CRL による失効情報	任意選択		**A.6.9**		
・・基本 OCSP 応答	任意選択		**A.6.9**		
・・他の失効情報	要別途規定		**A.6.9**		
・・CA 等による失効情報の保管	要別途規定		**A.6.9**		
CAdES-C データへのタイムスタンプ (CAdES-C-timestamp)	要別途規定		**A.6.10**		
タイムスタンプが付与された証明書及び失効情報に関する参照情報	要別途規定		**A.6.11**		
アーカイブタイムスタンプ id-aa-48	任意選択 a)		**A.6.12**		
アーカイブタイムスタンプ id-aa-27	任意選択 a)		**A.6.12**		
注 a) アーカイブタイムスタンプ id-aa-48 又はアーカイブタイムスタンプ id-aa 27 のどちらか又は両方が存在しなければならない。					

F.7　要別途規定要素が参照する仕様

表 F.2 から**表 F.6** で”要別途規定”の要素に✔を記入した場合は，その要素名と参照する仕様の名称を記入する

	要素名	参照する仕様の名称
1		
2		

F.8　特記事項

付録２　拡張可能なマーク付け言語を利用した電子署名(XAdES)の長期署名プロファイルに関する要求事項　JIS 案

本書は、今後の JIS 化審議過程において変更される可能性があります。

目　次

まえがき

この規格は，工業標準化法第 12 条第 1 項の規定に基づき，次世代電子商取引推進協議会(ECOM)から，工業標準原案を具して日本工業規格を制定すべきとの申出があり，日本工業標準調査会の審議を経て，経済産業大臣が制定した日本工業規格である。

この規格は，著作権法で保護対象となっている著作物である。

この規格の一部が，特許権，出願公開後の特許出願，実用新案権又は出願公開後の実用新案登録出願に抵触する可能性があることに注意を喚起する。経済産業大臣及び日本工業標準調査会は，このような特許権，出願公開後の特許出願，実用新案権又は出願公開後の実用新案登録出願に係る確認について，責任をもたない。

拡張可能なマーク付け言語を利用した電子署名(XAdES)の長期署名プロファイルに関する要求事項

Long term signature profiles for extensible markup language advanced electronic signatures(XAdES)

序文

この規格は，電子署名を長期にわたって検証可能にする長期署名に関する実装間の相互運用性を確保することを目的とする。

1　適用範囲

この規格は，署名プロファイルのうち，拡張可能なマーク付け言語を利用した電子署名のプロファイルに関する要求事項について規定する。

2　引用規格

次に掲げる規格は，この規格に引用されることによって，この規格の規程の一部を構成する。これらの引用規格は，その最新版（追補を含む。）を適用する。

JIS X 0001　情報処理用語－基本用語

JIS X 0008　情報処理用語－セキュリティ

3　用語及び定義

この規格で用いる主な用語及び定義は，**JIS X 0001** 及び **JIS X 0008** によるほか，次による。

3.1

長期署名(long term signature)

署名時刻の特定並びに署名対象及び検証情報を含む署名に関する情報の改ざん(竄)検知を可能とする措置を実施し，署名を長期にわたって検証可能にした署名。

3.2

プロファイル(profile)

参照する仕様の選択要素や値の範囲に関する相互運用性要件。

3.3

要求レベル(required level)

プロファイルを構成する各要素の実装に対する要求の程度。

3.4

拡張可能なマーク付け言語署名，XML 署名（XML signature)

任意のメッセージ内容に対する署名構文と処理。

3.5

拡張可能なマーク付け言語を利用した電子署名，XAdES (XML advanced electronic signatures)

署名者の識別やデータの改ざん(竄)検知が可能な XML 署名に基づく電子署名。

3.6

拡張可能なマーク付け言語を利用した時刻付き署名，XAdES-T(XAdES with time)

信頼のおける署名時刻（例えば，署名タイムスタンプ）を伴う XML 署名に基づく電子署名。

3.7

拡張可能なマーク付け言語を利用した長期保存署名，XAdES-A(archival XAdES)

署名対象及び検証情報を含む署名に関する情報の改ざん(竄)検知が可能な措置（例えば，アーカイブタイムスタンプ）を伴う XML 署名に基づく電子署名。

3.8

検証情報(validation data)

署名及びタイムスタンプの検証に用いる証明書及び失効情報。

3.9

タイムスタンピング機関，TSA(time-stamping authority)

あるデータが時間軸上のある点より前に存在していた証拠を提供することを信託された信頼できる第三者機関。

3.10

タイムスタンプトークン，TST(time-stamp token)

データの表現と時刻の値との間の，検証可能な暗号化された結合を含むデータ構造。タイムスタンプトークンは，その結合の中に追加のデータを含んでよい。

3.11

署名タイムスタンプ(signature timestamp)

署名が存在した時刻を特定可能にするために，署名値(SignatureValue)に付されるタイムスタンプ

3.12

アーカイブタイムスタンプ(archive timestamp)

改ざん(竄)を検知可能にするために，署名対象及び検証情報を含む署名に関する情報に付されるタイムスタンプ。

3.13

信頼点(trust anchor)

電子署名の検証を行う際の検証者によって信頼された信用の起点。公開かぎ(鍵)証明書または公開かぎ(鍵)の形式で提供され，一般的には，信頼する最上位の証明機関の公開かぎ(鍵)証明書である。

3.14

信頼できる第三者機関，TTP (trusted third party)

セキュリティ関連の活動に関して,他のエンティティから信頼されるセキュリティ機関又はその代理者。

3.15

証明機関，CA (certificate authority)

公開かぎ(鍵)証明書の作成及び割り当てを信託されたセンタ。任意選択として，証明機関は，エンティティに対してかぎ(鍵)を作成し，割り当てることもできる。

3.16

証明書(certificate)

あるエンティティの非対称かぎ(鍵)の一対のうち，公開されるかぎ(鍵)の情報で，証明機関によって署名され，それによって偽造できないようにしたもの。

3.17

属性証明書(attribute certificate)

職責，資格，地位などの属性及び属性値を記載した証明書。

3.18

失効情報(revocation information)

有効期間内に失効した証明書に関して証明機関が発行する情報。この情報と照合することで，証明書が現在も有効であるか否かを検証できる。

3.19

拡張セキュリティサービス，ESS (enhanced security service)

署名者証明書を特定する情報、署名の種類を表す情報など署名に対する拡張サービスオプション。

4　規格適合性

この規格に適合する次の実装 **a)**～**d)**は，この規格が規定する XAdES-T プロファイル又は XAdES-A プロファイルにおける要求レベルが“必す(須)”の要素の処理をすべて備え，かつ，要求レベルが“要別途規定”の要素の処理を備えている場合は，その処理に関する詳細な仕様が定義されていなければならない。

a) XAdES-T データの生成

b) XAdES-T データの検証

c) XAdES-A データの生成

d) XAdES-A データの検証

規格適合性の宣言は，実装者自らが，実装状況（及び“要別途規定”要素に関してはその仕様）をプロファイル適合性宣言**(附属書 F)**に記載しこれを開示することによって行う。

注記　XAdES-T データの生成及び検証，並びに XAdES-A データの生成及び検証の位置付けについては**図 1** 参照。

5　長期署名プロファイル

5.1　定義する長期署名プロファイル

電子署名を長期にわたって検証可能にするためには，相互運用性が確保されていることのほかに，署名時刻の特定が可能であることに加え，署名対象及び検証情報を含む署名に関する情報の改ざん(竄)検出が可能であることが必要である。この規格では，XAdES に関して，次の二つのプロファイルを定義することによって，これらの要求を満たす。

a)　XAdES-T プロファイル　XAdES-T データの生成及び検証に関するプロファイル。

b)　XAdES-A プロファイル　XAdES-A データの生成及び検証に関するプロファイル。

ここで，XAdES-T データと XAdES-A データの関係を**図 1** に示す。

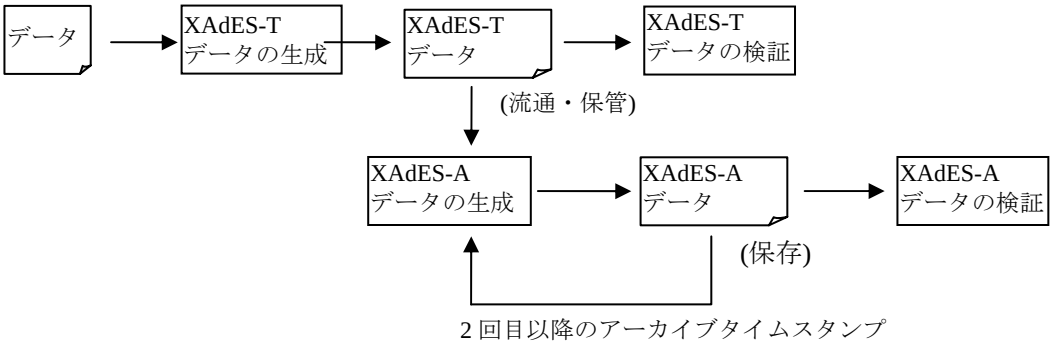


図 1－XAdES-T データと XAdES-A データとの関係

5.2　要求レベルの標語

この規格では，XAdES-T データ及び XAdES-A データを構成する各要素に対する，プロファイルとしての要求レベルとして，次の標語を定義する。

a)　必す(須)　この要求レベルをもつ要素は必ず実装しなければならない。この要求レベルの要素が，選択肢となる下位要素をもつ場合は，少なくともその一つを選択しなければならない。

b)　任意選択　この要求レベルをもつ要素の実装は任意である。

c)　要別途規定　この要求レベルをもつ要素の実装は任意であるが，その処理に関して，別途に詳細な仕様を規定しなければならない。

5.3　要求レベルの設定基準

XAdES-T 又は XAdES-A データを構成する各要素に対する要求レベルの設定は，次の基準に基づく。

a)　XAdES の定義で“必す(須)”となっている要素，並びに長期署名の生成及び検証に最低限必要な要素は“必す(須)”とする。

b)　外部定義要素は，“要別途規定”とする。

　　例　　その他形式の証明書

c)　特定アプリケーションとの連携を前提とした要素は，“要別途規定”とする。

　　例　　データオブジェクト形式

d)　運用に依存する要因が残っている要素は，“要別途規定”とする。

例　属性証明書系要素，タイムマークなど

注記　**ISO/IEC 1804-2** で定義されたアーカイビング方式のタイムスタンプはその他タイムスタンプに含まれる。

e) 単なる参考情報を運ぶ要素に関しては，その要素を実装していない場合，要求を無視する。

例　署名時刻

5.4　XAdES-T プロファイル

5.4.1　署名要素及びかぎ（鍵）情報要素

XML 署名で定義された署名要素及びかぎ（鍵）情報要素を構成する各要素は，**表 1** 及び**表 2** に示す要求レベルに従う。この表に定義されていない要素の要求レベルは“要別途規定”である。

表 1－署名要素

要素／属性	要求レベル	条件	参照箇条
ID	必す(須) [a)]		**A.2.1**
署名に関する情報	必す(須)		**A.2.2**
・・正規化方式	必す(須)		**A.2.2.1**
・・署名方式	必す(須)		**A.2.2.2**
・・参照情報	必す(須)		**A.2.2.3**
・・・・変換処理	任意選択		**A.2.2.3.1**
・・・・ダイジェスト方式	必す(須)		**A.2.2.3.2**
・・・・ダイジェスト値	必す(須)		**A.2.2.3.3**
署名値	必す(須)		**A.2.3**
かぎ(鍵)情報	任意選択 [b)]		**A.2.4**
オブジェクト	必す(須)		**A.2.5**

注記　イタリック体は属性を示す。

注 [a)]　XML 署名では“任意選択”であるが、XAdES では“必す(須)”。

[b)]　かぎ(鍵)情報又は署名者証明書の参照情報（**表 4**）のどちらか一方が必要。

表 2－かぎ（鍵）情報要素

要素	要求レベル	条件	参照箇条
署名かぎ(鍵)の名前	要別途規定		**A.3.1**
署名かぎ(鍵)の値	要別途規定		**A.3.2**
・・DSA かぎ(鍵)の値	要別途規定		**A.3.2**
・・RSA かぎ(鍵)の値	要別途規定		**A.3.2**
取得方式	要別途規定		**A.3.3**
X.509 データ	必す(須)		**A.3.4**
PGP データ	要別途規定		**A.3.5**
簡易 PKI データ	要別途規定		**A.3.6**
マネジメントデータ	要別途規定		**A.3.7**

5.4.2　オブジェクト要素，署名対象及び非署名対象プロパティ要素

XAdES で定義されたオブジェクト要素，署名対象プロパティ要素及び非署名対象プロパティ要素を構成する各要素は，**表 3**，**表 4** 及び**表 5** に示す要求レベルに従う。この表に定義されていない要素の要求レベルは“要別途規定”である。

表 3－オブジェクト要素

要素	要求レベル	条件	参照箇条
署名を修飾するプロパティ	必す(須)	対象属性に，署名要素の ID 属性の値を入れる	**A.4.1**
・・署名対象プロパティ	必す(須)		**A.4.1.1**
・・非署名対象プロパティ	任意選択		**A.4.1.2**
署名を修飾するプロパティを特定する参照情報	要別途規定		**A.4.2**

表 4－署名対象プロパティ要素

要素	要求レベル	条件	参照箇条
署名対象の署名プロパティ	必す(須)		**A.5.1**
・・署名時刻	任意選択 a)		**A.5.1.1**
・・署名者証明書の参照情報	任意選択 b)		**A.5.1.2**
・・署名ポリシ識別子	要別途規定		**A.5.1.3**
・・署名生成場所	要別途規定		**A.5.1.4**
・・署名者の肩書き	要別途規定		**A.5.1.5**
署名対象データオブジェクトのプロパティ	要別途規定		**A.5.2**
・・データオブジェクト形式	要別途規定		**A.5.2.1**
・・コミットメント種別表示	要別途規定		**A.5.2.2**
・・全データオブジェクトに対するタイムスタンプ	要別途規定		**A.5.2.3**
・・個別データオブジェクトに対するタイムスタンプ	要別途規定		**A.5.2.4**
注 a) 未実装の場合は無視。 b) 署名者証明書の参照情報か，かぎ(鍵)情報（**表 2**）のどちらか一方が必要。			

表 5－非署名対象プロパティ要素

要素	要求レベル	条件	参照箇条
非署名対象の署名プロパティ	必す(須)		**6.1**
・・カウンタ署名	任意選択		**6.1.1**
・・署名タイムスタンプ	任意選択 a)		**6.1.2**
・・タイムマークなどその他の方式	要別途規定 a)		**6.1.3**
非署名のデータオブジェクトのプロパティ	要別途規定		**6.2**
注 a) 署名タイムスタンプかタイムマークなどその他の方式のいずれか一方を選択。			

5.5 XAdES-A プロファイル

5.5.1 前提条件

XAdES-A プロファイルは，XAdES-T データの拡張として定義される。基となる XAdES-T データは，必ずしも **5.4** の XAdES-T プロファイルに準じる必要はない。

5.5.2 非署名対象の署名プロパティ要素

XAdES で定義された，XAdES-T に追加される非署名対象の署名プロパティ要素の各要素は，**表 6** に示す要求レベルに従う。この表に定義されていない要素の要求レベルは“要別途規定”である。

表 6　非署名対象の署名プロパティ要素

要素／処理方式	要求レベル	条件	参照箇条
全証明書参照情報群	任意選択		**A.7.1**
全失効情報参照情報群	任意選択		**A.7.2**
・・CRL 形式の失効情報参照情報	任意選択		**A.7.2**
・・OCSP 形式の失効情報参照情報	任意選択		**A.7.2**
・・他の失効情報参照情報	要別途規定		**A.7.2**
属性証明書参照情報群	要別途規定		**A.7.3**
属性失効情報参照情報群	要別途規定		**A.7.4**
署名及び参照情報に対するタイムスタンプ	任意選択 **a) b)**		**A.7.5**
・・*非分離型*	必す(須)		**A.7.5**
・・*分離型*	要別途規定		**A.7.5**
参照情報に対するタイムスタンプ	任意選択 **a) b)**		**A.7.6**
・・*非分離型*	必す(須)		**A.7.6**
・・*分離型*	要別途規定		**A.7.6**
証明書群	必す(須)		**A.7.7**
・・カプセル構造化された証明書	任意選択		**A.7.7**
・・他の証明書	要別途規定		**A.7.7**
・・CA 等による証明書の保管	要別途規定		**A.7.7**
失効情報群	必す(須)		**A.7.8**
・・CRL による失効情報群	任意選択		**A.7.8**
・・OCSP による失効情報群	任意選択		**A.7.8**
・・他の失効情報群	要別途規定		**A.7.8**
・・CA 等による失効情報の保管	要別途規定		**A.7.8**
属性証明書群	要別途規定		**A.7.9**
属性失効情報群	要別途規定		**A.7.10**
アーカイブタイムスタンプ	必す(須)		**A.7.11**
・・*非分離型*	必す(須)		**A.7.11**
・・*分離型*	要別途規定		**A.7.11**
異なる版の非署名対象の署名プロパティ	要別途規定		**A.7.12**

注記　イタリック体は処理方式を示す。
注 **a)**　選択する場合は，署名及び参照情報に対するタイムスタンプか参照情報に対するタイムスタンプのいずれか一方だけ可能。
b)　推奨しない。

5.6　タイムスタンプの TSA 証明書検証情報

過去のタイムスタンプの TSA 証明書を検証する際には, 信頼点までの証明書及び失効情報で構成される証明書検証情報が必要となる。TSA 証明書の場合には，TSA の証明書から信頼点の証明書までの証明書チェーン上の証明書と、それぞれの失効情報となる。

検証情報は、次に示す方法によって XAdES-A データ内に格納することができる。XAdES-A データ内に格納しない場合は、他の安全な手段によって保存されている必要がある。他の安全な手段の例は，TTP である CA 又は TSA が保存しておく方法である。

また、過去のタイムスタンプの検証の際には、次に示す方法によって格納された検証情報を利用してもよいし、他の安全な手段によって保存された検証情報を利用してもよい。

タイムスタンプトークンの構造に関しては，要求事項を**附属書 E** に示す。

a)　署名タイムスタンプの TSA 証明書検証情報　生成時，次のいずれかの場所に格納するか，又は CA 等で保存する。

1) 非署名属性内の次の要素。

－証明書群

－失効情報群

2) 署名タイムスタンプにおけるタイムスタンプトークン（署名タイムスタンプトークン）内の次の要素。

－証明書群

－失効情報群

3) 署名タイムスタンプトークンの非署名属性群内の次の要素。

－証明書群

－失効情報群

b) アーカイブタイムスタンプの TSA 証明書検証情報　生成時，次のいずれかの場所に格納するか，又は CA 等で保存する。

1) アーカイブタイムスタンプにおけるタイムスタンプトークン（アーカイブタイムスタンプトークン）内の次の要素。

－証明書群

－失効情報群

2) アーカイブタイムスタンプトークンの非署名属性群内の次の要素。

－証明書群

－失効情報群

附属書 A
（規定）
XAdES のデータ構造と構成要素

序文

この附属書は，XAdES のデータ構造と構成要素を規定する。

A.1　XAdES データの構造

XAdES データの構造は，XML 署名の拡張形として定義される。XML 署名の定義に従って，“署名”要素が最上位要素となる**図 A.1** の基本構造をもつ。**図 A.2** に“オブジェクト”要素の構造を示す。また，“署名対象プロパティ”要素の構造を**図 A.3** に，“非署名対象プロパティ”要素の構造を**図 A.4** に示す。

<table>
<tr><td rowspan="12">署名</td><td rowspan="4">署名に関する情報</td><td colspan="2">正規化方式</td></tr>
<tr><td colspan="2">署名方式</td></tr>
<tr><td rowspan="3">コンテントへの参照情報</td><td>変換処理</td></tr>
<tr><td>ダイジェスト方式</td></tr>
<tr><td>ダイジェスト値</td></tr>
<tr><td colspan="2">署名値</td></tr>
<tr><td rowspan="7">かぎ(鍵)情報</td><td colspan="2">署名かぎ(鍵)の名前</td></tr>
<tr><td rowspan="2">署名かぎ(鍵)の値</td><td>DSA かぎ(鍵)の値</td></tr>
<tr><td>RSA かぎ(鍵)の値</td></tr>
<tr><td colspan="2">取得方式</td></tr>
<tr><td colspan="2">X.509 データ</td></tr>
<tr><td colspan="2">PGP データ</td></tr>
<tr><td colspan="2">簡易 PKI データ</td></tr>
<tr><td colspan="2">マネジメントデータ</td></tr>
<tr><td colspan="2">オブジェクト</td></tr>
</table>

図 A.1－XAdES データの基本構造

<table>
<tr><td rowspan="2">オブジェクト</td><td rowspan="2">署名を修飾するプロパティ</td><td>署名対象プロパティ</td></tr>
<tr><td>非署名対象プロパティ</td></tr>
<tr><td colspan="2">署名を修飾するプロパティを特定する参照情報</td></tr>
</table>

図 A.2－オブジェクトの構造

<table>
<tr><td rowspan="9">署名対象プロパティ</td><td rowspan="5">署名対象の署名プロパティ</td><td>署名時刻</td></tr>
<tr><td>署名者証明書</td></tr>
<tr><td>署名ポリシ識別子</td></tr>
<tr><td>署名生成場所</td></tr>
<tr><td>署名者の肩書き</td></tr>
<tr><td rowspan="4">署名対象データオブジェクトのプロパティ</td><td>データオブジェクト形式</td></tr>
<tr><td>コミットメント種別表示</td></tr>
<tr><td>全データオブジェクトに対するタイムスタンプ</td></tr>
<tr><td>個別データオブジェクトに対するタイムスタンプ</td></tr>
</table>

図 A.3－署名対象プロパティ

<table>
<tr><td rowspan="20">非署名対象プロパティ</td><td rowspan="19">非署名対象の署名プロパティ</td><td colspan="2">カウンタ署名</td></tr>
<tr><td colspan="2">署名タイムスタンプ</td></tr>
<tr><td colspan="2">タイムマークなどその他の方式</td></tr>
<tr><td colspan="2">全証明書参照情報群</td></tr>
<tr><td rowspan="3">全失効情報参照情報群</td><td>CRL 形式の失効情報参照情報</td></tr>
<tr><td>OCSP 形式の失効情報参照情報</td></tr>
<tr><td>他の失効情報参照情報</td></tr>
<tr><td colspan="2">属性証明書参照情報群</td></tr>
<tr><td colspan="2">属性失効情報参照情報群</td></tr>
<tr><td colspan="2">署名及び参照情報に対するタイムスタンプ</td></tr>
<tr><td colspan="2">参照情報に対するタイムスタンプ</td></tr>
<tr><td rowspan="3">証明書群</td><td>カプセル構造化された証明書</td></tr>
<tr><td>他の証明書</td></tr>
<tr><td>CA 等による証明書の保管</td></tr>
<tr><td rowspan="4">失効情報群</td><td>CRL による失効情報群</td></tr>
<tr><td>OCSP による失効情報群</td></tr>
<tr><td>他の失効情報群</td></tr>
<tr><td>CA 等による失効情報の保管</td></tr>
<tr><td colspan="2">属性証明書群</td></tr>
<tr><td colspan="2">属性失効情報群</td></tr>
<tr><td colspan="2">アーカイブタイムスタンプ</td></tr>
<tr><td colspan="2">異なる版の非署名対象の署名プロパティ</td></tr>
<tr><td colspan="2">非署名のデータオブジェクトのプロパティ</td></tr>
</table>

図 A.4－非署名対象プロパティ

A.2　署名要素及び属性

A.2.1　ID 属性

署名要素における ID 属性。長期署名では必す(須)。

A.2.2　署名に関する情報

署名値を計算する際の入力となる署名対象情報を格納する。

A.2.2.1　正規化方式

同じ意味をもつ XML 文書がビット単位で同じ表現の文書になるようにする変換処理の方式。

A.2.2.2　署名方式

署名を生成または検証するときの署名アルゴリズム。

A.2.2.3　コンテントへの参照情報

署名対象となる要素またはデータの参照先、およびそのダイジェスト値を格納する。

A.2.2.3.1　変換処理

正規化などの変換処理。並べられた順番の通りに変換処理が行われる。

A.2.2.3.2　ダイジェスト方式

ダイジェストの方式。

A.2.2.3.3　ダイジェスト値

ダイジェストの値。この値は Base64 形式でエンコードされたものが使用される。

A.2.3　署名値

署名の値。

A.2.4　かぎ(鍵)情報

署名検証に使用されるかぎ(鍵)情報を格納する。

A.2.5　オブジェクト

任意のデータを含むことができる要素。XML 署名中に複数存在することもある。

A.3　かぎ(鍵)情報要素

A.3.1　署名かぎ(鍵)の名前

署名かぎ(鍵)の識別子。

A.3.2　署名かぎ(鍵)の値

署名かぎ(鍵)の値。

a)　DSA かぎ(鍵)の値

b)　RSA かぎ(鍵)の値

A.3.3　取得方式

文書外などにあるかぎ(鍵)情報の取得方式。

A.3.4　X.509 データ

１つまたは複数の X.509 証明書またはその識別子や CRL などの失効情報に関する情報を格納するための任意要素。

A.3.5　PGP データ

PGP 公開かぎ(鍵)識別子などを格納する。

A.3.6　簡易 PKI データ

簡易 PKI(SPKI，Simple Public Key Infrastructure）の証明書などを格納する。

A.3.7　マネジメントデータ

DH かぎ(鍵)などのかぎ(鍵)配送に使用する。

A.4　オブジェクト要素

A.4.1　署名を修飾するプロパティ

署名に必要な追加のプロパティを格納する。

A.4.1.1　署名対象プロパティ

署名対象となるプロパティ。署名に関するプロパティと署名以外のプロパティとから成る。署名値の計算対象となるよう、参照情報(A.2.2.3)で参照される。

A.4.1.2　非署名対象プロパティ

署名対象とはならないプロパティ。

A.4.2　署名を修飾するプロパティを特定する参照情報

署名署名に必要な追加のプロパティを特定する参照情報。

A.5　署名対象プロパティ要素

A.5.1　署名対象の署名プロパティ

署名対象となる署名に関するプロパティ。

A.5.1.1　署名時刻

署名者が主張する署名時刻。

A.5.1.2　署名者証明書

署名者の証明書。

A.5.1.3　署名ポリシ識別子

署名ポリシを特定するための情報。

A.5.1.4　署名生成場所

署名が生成された場所。第三者によって保証されたものではなく署名者が主張している場所。

A.5.1.5　署名者の肩書き

署名者が主張する署名者の肩書き，または属性証明書で保証された肩書き。

A.5.2　署名対象データオブジェクトのプロパティ

署名対象のうちの，データオブジェクト（文書）に関連するプロパティ。

A.5.2.1　データオブジェクト形式

データオブジェクト形式のヒント。

A.5.2.2　コミットメント種別表示

署名者がどのような意図を持って署名したかを表明する情報。

A.5.2.3　全データオブジェクトに対するタイムスタンプ

署名生成前の署名対象に対するタイムスタンプ。

A.5.2.4　個別データオブジェクトに対するタイムスタンプ

署名生成前の特定の署名対象に対するタイムスタンプ。

A.6　非名対象プロパティ要素

A.6.1　非署名対象の署名プロパティ

署名対象外の署名に関するプロパティ。

A.6.1.1　カウンタ署名

署名値に対する署名。複数人が同一文書に署名する際、署名の順序に意味がある、もしくは意味を持たせたい場合に用いる。

A.6.1.2　署名タイムスタンプ

署名が存在した時刻を特定可能にするために，署名値に付されるタイムスタンプ。

A.6.1.3　タイムマークなどその他の方式

データが特定の時刻以前に存在したことを示すための，データと特定の時間を結びつける信頼される第三者機関からの監査証跡内の情報など。

A.6.2　非署名のデータオブジェクトのプロパティ

データオブジェクト（文書）に関連する署名対象とはならないプロパティ。

A.7.　非署名対象の署名プロパティ要素

A.7.1　全証明書参照情報群

署名者を除く証明書チェーンを構成する全ての証明書参照情報のかたまり。

A.7.2　全失効情報参照情報群

署名者証明書の証明書チェーンの検証に必要な全ての失効情報参照情報のかたまり。

a)　CRL 形式の失効情報参照情報

b）OCSP 形式の失効情報参照情報

c) 他の失効情報参照情報

A.7.3 属性証明書参照情報群

関連する属性証明書に関する参照情報のかたまり。

A.7.4 属性失効情報参照情報群

関連する属性証明書の失効情報に関する参照情報のかたまり。

A.7.5 署名及び参照情報に対するタイムスタンプ

署名及び参照情報に対するタイムスタンプ。非分離型と分離型が定義されている。

7.6 参照情報に対するタイムスタンプ

参照情報に対するタイムスタンプ。非分離型と分離型が定義されている。

7.7 証明書群

署名者証明書の証明書チェーンを構成する全ての証明書のかたまり。

a) カプセル構造化された証明書

b) 他の証明書

c) CA 等による証明書の保管

7.8 失効情報群

署名者証明書の証明書チェーン検証に必要な全ての失効情報のかたまり。

a) CRL による失効情報群

b) OCSP による失効情報群

c) 他の失効情報群

d) CA 等による失効情報の保管

7.9 属性証明書群

関連する属性証明書のかたまり。

7.10 属性失効情報群

関連する属性証明書に対する失効情報のかたまり。

7.11 アーカイブタイムスタンプ

改ざん(竄)を検知可能にするために，署名対象及び検証情報を含む署名に関する情報に付されるタイムスタンプ。非分離型と分離型が定義されている。

7.12 異なる版の非署名対象の署名プロパティ

異なる版の XAdES で定義された署名対象とはならない署名プロパティ。

附属書 B
（参考）
参照文献

序文

この附属書は，長期署名プロファイルに関する要求事項に沿った実装が参照する仕様について記載するものであって，規定の一部ではない。

B.1　XAdES 仕様

ETSI TS 101 903 XML Advanced Electronic Signatures (XAdES)

注記 1　http://pda.etsi.org/pda/queryform.asp から入手可能。

2　v1.1.1 は v1.3.2 と次の点が異なる。

- かぎ(鍵)情報はなくてもよい。
- 署名時刻，署名者証明書の参照情報，全証明書参照情報群及び全失効情報参照情報群は“必す(須)”。
- 属性証明書群，属性失効情報群，属性証明書参照情報群及び属性失効情報参照情報群，並びに非分離型及び分離型は未定義。
- 署名及び参照情報に対するタイムスタンプか参照情報に対するタイムスタンプのいずれか一方が必要。

3　v1.2.2 は v1.3.2 と次の点が異なる。

- 属性証明書群及び属性失効情報群並びに非分離型及び分離型は未定義。

B.2　XML 署名仕様

XML-Signature Syntax and Processing W3C Recommendation 12 February 2002

注記　http://www.w3.org/TR/xmldsig-core/から入手可能。

B.3　CAdES 仕様

ETSI TS 101 733 CMS Advanced Electronic Signatures (CAdES)

注記　http://pda.etsi.org/pda/queryform.asp から入手可能。

B.4　CMS 仕様

IETF RFC 3852 Cryptographic Message Syntax

注記　http://www.ietf.org/rfc.html から入手可能。

附属書 C
（参考）
要素名及び属性名と XML 表記の対応表

序文

この附属書は，この規格で用いられる要素及び属性の XML 表記について記載するものであって，規定の一部ではない。

表 C.1－署名要素(ds:Signature element)

要素／属性	XML 表記
ID	*ID*
署名に関する情報	ds:SignedInfo
・・正規化方式	・・ds:CanonicalizationMethod
・・署名方式	・・ds:SignatureMethod
・・コンテントへの参照情報	・・ds:Reference
・・・・変換処理	・・・・ds:Transforms
・・・・ダイジェスト方式	・・・・ds:DigestMethod
・・・・ダイジェスト値	・・・・ds:DigestValue
署名値	ds:SignatureValue
かぎ(鍵)情報	ds:KeyInfo
オブジェクト	ds:Object

表 C.2－かぎ（鍵）情報要素(ds:KeyInfo element)

要素	XML 表記
署名かぎ(鍵)の名前	ds:KeyName
署名かぎ(鍵)の値	ds:KeyValue
・・DSA かぎ(鍵)の値	・・ds:DSAKeyValue
・・RSA かぎ(鍵)の値	・・ds:RSAKeyValue
取得方式	ds:RetrievalMethod
X.509 データ	ds:X509Data
PGP データ	ds:PGPData
簡易 PKI データ	ds:SPKIData
マネジメントデータ	ds:MgmtData

表 C.3－オブジェクト要素(ds:Object element)

要素	XML 表記
署名を修飾するプロパティ	QualifyingProperties
・・署名対象プロパティ	・・SignedProperties
・・非署名対象プロパティ	・・UnsignedProperties
署名を修飾するプロパティを特定する参照情報	QualifyingPropertiesReference

表 C.4－署名対象プロパティ要素(SignedProperties element)

要素	XML 表記
署名対象の署名プロパティ	SignedSignatureProperties
・・署名時刻	・・SigningTime
・・署名者証明書の参照情報	・・SigningCertificate
・・署名ポリシ識別子	・・SignaturePolicyIdentifier
・・署名生成場所	・・SignatureProductionPlace
・・署名者の肩書き	・・SignerRole
署名対象データオブジェクトのプロパティ	SignedDataObjectProperties
・・データオブジェクト形式	・・DataObjectFormat
・・コミットメント種別表示	・・CommitmentTypeIndication
・・全データオブジェクトに対するタイムスタンプ	・・AllDataObjectsTimeStamp
・・個別データオブジェクトに対するタイムスタンプ	

表 C.5－非署名対象プロパティ要素(UnsignedProperties element)

要素	XML 表記
非署名対象の署名プロパティ	UnsignedSignatureProperties
・・カウンタ署名	・・CounterSignature
・・署名タイムスタンプ	・・SignatureTimeStamp
・・タイムマークなどその他の方式	・・
非署名のデータオブジェクトのプロパティ	UnsignedDataObjectProperties

表 C.6　非署名対象の署名プロパティ要素(UnsignedSignatureProperties element)

要素／処理方式	XML 表記
全証明書参照情報群	CompleteCertificateRefs
全失効情報参照情報群	CompleteRevocationRefs
・・CRL 形式の失効情報参照情報	・・CRLRef
・・OCSP 形式の失効情報参照情報	・・OCSPRef
・・他の失効情報参照情報	・・OtherRef
属性証明書参照情報群	AttributeCetrificateRefs
属性失効情報参照情報群	AttributeRevocationRefs
署名及び参照情報に対するタイムスタンプ	SigAndRefsTimeStamp
・・非分離型	・・not distributed case
・・分離型	・・distributed case
参照情報に対するタイムスタンプ	RefsOnlyTimeStamp
・・非分離型	・・not distributed case
・・分離型	・・distributed case
証明書群	CertificateValues
・・カプセル構造化された証明書	・・EncapsulatedX509Certificate
・・他の証明書	・・OtherCertificate
・・CA 等による証明書の保管	・・
失効情報群	RevocationValues
・・CRL による失効情報群	・・CRLValues
・・OCSP による失効情報群	・・OCSPValues
・・他の失効情報群	・・OtherValues
・・CA 等による失効情報の保管	・・
属性証明書群	AttrAuthoritiesCertValues
属性失効情報群	AttributeRevocationValues
アーカイブタイムスタンプ	ArchiveTimeStamp
・・非分離型	・・not distributed case
・・分離型	・・distributed case
異なる版の非署名対象の署名プロパティ要素	Any unsigned signature property defined in any other version of XAdES

附属書 D
（規定）
タイムスタンプトークンの構造

序文

この附属書は，長期署名におけるタイムスタンプトークンの構造に関する要求事項を規定する。

D.1　準拠する仕様

この規格における署名タイムスタンプトークン及びアーカイブタイムスタンプトークンは次の仕様に準拠する。

a)　基本構造　CMS に準拠する。

b)　署名属性及び非署名属性　CAdES に準拠する。

D.2　構成要素の要求レベル

署名タイムスタンプトークン及びアーカイブタイムスタンプトークンの各要素に対する要求レベルは**表 D.1** に従う。

表 D.1－タイムスタンプトークンの各要素に対する要求レベル

要素	要求レベル	条件又は値
コンテント種別	必す(須)	署名付きデータを表す識別子
コンテント	必す(須)	署名付きデータ
・・暗号メッセージ構文の版数	必す(須)	
・・ダイジェストアルゴリズム識別子群	必す(須)	
・・カプセル構造化されたコンテント情報	必す(須)	
・・・・e コンテント種別	必す(須)	TST 情報を表す識別子
・・・・e コンテント	必す(須)	TST 情報
・・証明書群	任意選択	
・・・・証明書	任意選択	
・・・・属性証明書 1 版	任意選択	
・・・・属性証明書 2 版	任意選択	
・・・・その他形式の証明書	要別途規定	
・・失効情報群	任意選択	
・・・・失効情報	任意選択	
・・・・その他形式の失効情報	任意選択	
・・署名者情報群	必す(須)	

表 D.1－タイムスタンプトークンの各要素に対する要求レベル（続き）

要素	要求レベル	条件又は値
・・・・署名者情報	必す(須)	
・・・・・・暗号メッセージ構文の版数	必す(須)	
・・・・・・署名者識別子	必す(須)	
・・・・・・・・発行者及びシリアル番号	任意選択	
・・・・・・・・対象者かぎ（鍵）識別子	任意選択	
・・・・・・ダイジェストアルゴリズム識別子	必す(須)	
・・・・・・署名属性群	必す(須)	
・・・・・・・・コンテント種別	必す(須)	TST 情報を表す識別子
・・・・・・・・メッセージダイジェスト	必す(須)	
・・・・・・・・署名者証明書の参照情報	必す(須)	
・・・・・・・・・・ESS 署名者証明書の参照情報	任意選択 a)	
・・・・・・・・・・ESS 署名者証明書の参照情報 2 版	任意選択 a)	
・・・・・・・・・・他の署名者証明書の参照情報	任意選択 a)	
・・・・・・署名アルゴリズム識別子	必す(須)	
・・・・・・署名値	必す(須)	
・・・・・・非署名属性群	任意選択	
・・・・・・・・完全な証明書参照情報群	任意選択	
・・・・・・・・完全な失効参照情報群	任意選択	
・・・・・・・・・・CRL 形式の失効参照情報群	任意選択	
・・・・・・・・・・OCSP 形式の失効参照情報群	任意選択	
・・・・・・・・証明書群	任意選択	
・・・・・・・・・・証明書	任意選択	
・・・・・・・・・・CA 等による証明書の保管	要別途規定	
・・・・・・・・失効情報群	任意選択	
・・・・・・・・・・CRL による失効情報	任意選択	
・・・・・・・・・・基本 OCSP 応答	任意選択	
・・・・・・・・・・他の失効情報	要別途規定	
注 a) ESS 署名者証明書の参照情報，ESS 署名者証明書の参照情報 2 版，他の署名者証明書の参照情報のいずれか一つを選択。		

附属書 E
（参考）
PDF/A への長期署名の適用方法

序文

ISO 19005-1 として標準化された、長期保存を目的とした PDF/A は、電子署名を付与することが可能であるが、現時点においては，署名ハンドラ及び符号化方法に対する名前付け規則が決まっておらず，また，アーカイブタイムタイムスタンプを用いた長期署名において、電子署名データのサイズ増加に伴う署名対象範囲の扱い方，及び可変長署名データの格納方法が標準化されておらず，実装依存となることにより相互運用性が確保できないことが懸念される。

このため、この附属書では，相互運用性確保に必要な名前付け規則，署名対象範囲の指定方法及び署名データの格納方法について参考のために記載するものであって，規定の一部ではない。

E.1　フィルタ名及びサブフィルタ名の名前付け規則

a)　名前の基本構造は，次の通りとする。

“商標登録されているなど一意性のある製造者名”＋“ドット”＋“製造者名のもとで一意な名前”

例　ecom . example

b)　この附属書では，PDF/A への長期署名適用のために，次の名前を定義する。

1)　**.AdES**　長期署名のための署名ハンドラの名前（フィルタ名）

2)　**jis.cades**　CAdES 処理を示す名前（サブフィルタ名）

3)　**jis.xades**　XAdES 処理を示す名前（サブフィルタ名）

E.2　バイトレンジの指定方法

署名範囲を示すバイトレンジは**表 E.1** に従う。

注記　**図 E.1** のように，時刻付き署名データ部分を避けて①及び②の範囲を指定する。

表 E.1 - バイトレンジ

配列中の位置	値	注記
0	ファイルの先頭	ゼロ
1	配列 0 の位置から署名タグ内の Contents の実データ直前までのバイト長さ	“<”までの長さ
2	署名タグ内の Contents の実データ直後のバイトオフセット	“>”の位置
3	配列 2 の位置から長期保存署名データ（添付ファイル）の直前までのバイト長さ	

E.3　時刻付き署名データ及び長期保存署名データの格納方法

時刻付き署名データを PDF コンテンツの署名タグ内に格納し，可変長の長期保存署名データは添付ファイルとして格納する（**図 E.1** 参照）。

a) 署名タグ内に“/sigAdES”を記述し，長期保存署名データを格納した添付ファイルの存在を表示する。

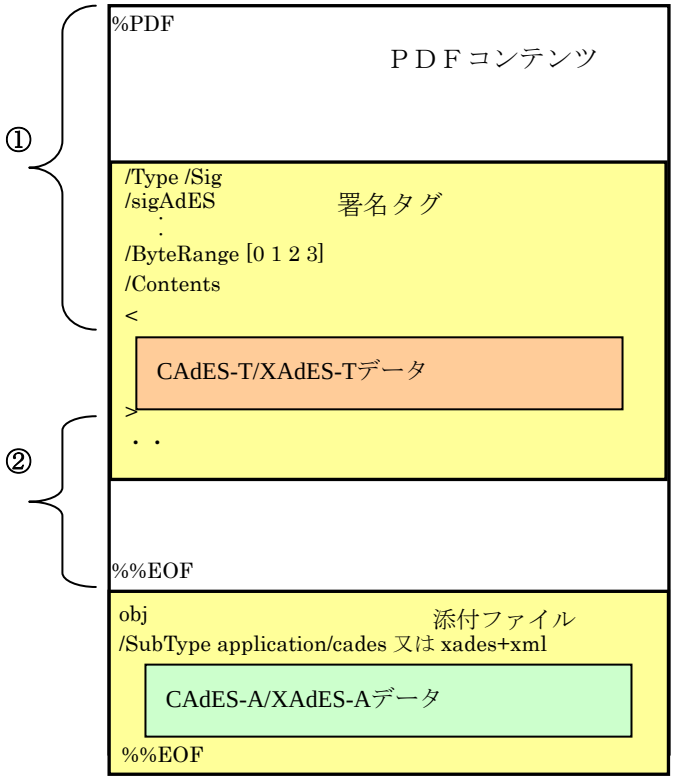
b) 添付ファイルの MIME タイプは“application/cades”又は“application/xades+xml”とする。

c) 署名タグ内の時刻付き署名データと，添付ファイルに格納する長期保存署名データ内の時刻付き署名データ部分は同一とする。

注記　署名タグ内に時刻付き署名データを格納するタイミングで，同時に，添付ファイル内にも同じ時刻付き署名データを格納する。

d) 添付ファイルに格納する長期保存署名データはバイナリ変換せず、そのまま格納する。

e) 長期署名の延長（アーカイブタイムスタンプの再取得）は，添付ファイルに格納された長期保存署名データに対して行い，添付ファイルを差し替える。



注記　図では，相互参照セクションやトレーラなどは省略している。

図 E.1 - PDF ファイルへのデータ格納方法

附属書 F
（規定）
プロファイル適合性宣言

序文

この附属書は，XAdES の実装に対するプロファイルの適合性を宣言するための書式を規定する。

記入方法

実装されている要素を“生成”及び“検証”欄にチェックマーク（✔）で示す。

F.1　記入者及び記入日

記入者	
記入日	

F.2　実装の名称

名称	
版番号	

F.3　XAdES のバージョン

バージョン番号	

F.4　プロファイル適合性を宣言する実装の範囲

表 F.1－実装範囲

	生成	検証
XAdES-T		
XAdES-A		

F.5　XAdES-T プロファイルへの適合

表 F.2－署名要素

要素／属性	要求レベル	条件	参照箇条	生成	検証
ID	必す(須)[a)]		**A.2.1**		
署名に関する情報	必す(須)		**A.2.2**		
・・正規化方式	必す(須)		**A.2.2.1**		
・・署名方式	必す(須)		**A.2.2.2**		
・・コンテントへの参照情報	必す(須)		**A.2.2.3**		
・・・・変換処理	任意選択		**A.2.2.3.1**		
・・・・ダイジェスト方式	必す(須)		**A.2.2.3.2**		
・・・・ダイジェスト値	必す(須)		**A.2.2.3.3**		
署名値	必す(須)		**A.2.3**		
かぎ(鍵)情報	任意選択 [b)]		**A.2.4**		
オブジェクト	必す(須)		**A.2.5**		

注記　イタリック体は属性を示す。

注 [a)]　XML 署名では“任意選択”と規定されるが、XAdES では“必す(須)”。

[b)]　かぎ(鍵)情報又は署名者証明書の参照情報（**表 F.5**）のどちらか一方が必要。

表 F.3－かぎ(鍵)情報要素

要素	要求レベル	条件	参照箇条	生成	検証
署名かぎ(鍵)の名前	要別途規定		**A.3.1**		
署名かぎ(鍵)の値	要別途規定		**A.3.2**		
・・DSA かぎ(鍵)の値	要別途規定		**A.3.2**		
・・RSA かぎ(鍵)の値	要別途規定		**A.3.2**		
取得方式	要別途規定		**A.3.3**		
X.509 データ	必す(須)		**A.3.4**		
PGP データ	要別途規定		**A.3.5**		
簡易 PKI データ	要別途規定		**A.3.6**		
マネジメントデータ	要別途規定		**A.3.7**		

表 F.4－オブジェクト要素

要素	要求レベル	条件	参照箇条	生成	検証
署名を修飾するプロパティ	必す(須)		**A.4.1**		
・・署名対象プロパティ	必す(須)		**A.4.1.1**		
・・非署名対象プロパティ	任意選択		**A.4.1.2**		
署名を修飾するプロパティを特定する参照情報	要別途規定		**A.4.2**		

表 F.5－署名対象プロパティ要素

要素	要求レベル	条件	参照箇条	生成	検証
署名対象の署名プロパティ	必す(須)		**A.5.1**		
・・署名時刻	任意選択 a)		**A.5.1.1**		
・・署名者証明書の参照情報	任意選択 b)		**A.5.1.2**		
・・署名ポリシ識別子	任意選択		**A.5.1.3**		
・・署名生成場所	任意選択		**A.5.1.4**		
・・署名者の肩書き	任意選択		**A.5.1.5**		
署名対象データオブジェクトのプロパティ	任意選択		**A.5.2**		
・・データオブジェクト形式	要別途規定		**A.5.2.1**		
・・コミットメント種別表示	要別途規定		**A.5.2.2**		
・・全データオブジェクトに対するタイムスタンプ	要別途規定		**A.5.2.3**		
・・個別データオブジェクトに対するタイムスタンプ	要別途規定		**A.5.2.4**		
注 a) 未実装の場合は無視。 b) 署名者証明書の参照情報か，かぎ(鍵)情報（**表 F.2**）のどちらか一方が必要。					

表 F.6－非署名対象プロパティ要素

要素	要求レベル	条件	参照箇条	生成	検証
非署名対象の署名プロパティ	必す(須)		**6.1**		
・・カウンタ署名	任意選択		**6.1.1**		
・・署名タイムスタンプ	任意選択 a)		**6.1.2**		
・・タイムマークなどその他の方式	要別途規定 a)		**6.1.3**		
非署名のデータオブジェクトのプロパティ	要別途規定		**6.2**		
注 a) 署名タイムスタンプかタイムマークなどその他の方式のいずれか一 方を選択。					

F.6　XAdES-A プロファイルへの適合

表 F.7－追加非署名対象の署名プロパティ要素

要素	要求レベル	条件	参照箇条	生成	検証
全証明書参照情報群	任意選択		**A.7.1**		
全失効情報参照情報群	任意選択		**A.7.2**		
・・CRL 形式の失効情報参照情報	任意選択		**A.7.2**		
・・OCSP 形式の失効情報参照情報	任意選択		**A.7.2**		
・・他の失効情報参照情報	要別途規定		**A.7.2**		
属性証明書参照情報群	要別途規定		**A.7.3**		
属性失効情報参照情報群	要別途規定		**A.7.4**		
署名及び参照情報に対するタイムスタンプ	任意選択 a) b)		**A.7.5**		
・・非分離型	必す(須)		**A.7.5**		
・・分離型	要別途規定		**A.7.5**		
参照情報に対するタイムスタンプ	任意選択 a) b)		**A.7.6**		
・・非分離型	必す(須)		**A.7.6**		
・・分離型	要別途規定		**A.7.6**		
証明書群	必す(須)		**A.7.7**		
・・カプセル構造化された証明書	任意選択		**A.7.7**		
・・他の証明書	要別途規定		**A.7.7**		
・・CA 等による証明書の保管	要別途規定		**A.7.7**		
失効情報群	必す(須)		**A.7.8**		
・・CRL による失効情報群	任意選択		**A.7.8**		
・・OCSP による失効情報群	任意選択		**A.7.8**		
・・他の失効情報群	要別途規定		**A.7.8**		
・・CA 等による失効情報の保管	要別途規定		**A.7.8**		
属性証明書群	要別途規定		**A.7.9**		
属性失効情報群	要別途規定		**A.7.10**		
アーカイブタイムスタンプ	必す(須)		**A.7.11**		
・・非分離型	必す(須)		**A.7.11**		
・・分離型	要別途規定		**A.7.11**		
異なる版の非署名対象の署名プロパティ要素	要別途規定		**A.7.12**		

注記　イタリック体は処理方式を示す

注 a)　選択する場合は，署名及び参照情報に対するタイムスタンプか参照情報に対するタイムスタンプのいずれか一方だけ可能。

b)　推奨しない。

F.7　要別途規定要素が参照する仕様

表 F.2 から**表 F.7** で”で”要別途規定”の要素に✔を記入した場合は，その要素名と参照する仕様の名称を記入する。

要素名		参照する仕様の名称
1		
2		

F.8　特記事項

メンバリスト

事務局

前田　陽二　　次世代電子商取引推進協議会（ECOM）

顧問

松本　　勉　　横浜国立大学　大学院

平田　健治　　大阪大学　大学院

米丸　恒治　　神戸大学

リーダ/幹事

リーダ　（SWG1 担当）木村　道弘　　日本電気株式会社

リーダ　（SWG2 担当）宮崎　一哉　　三菱電機株式会社

幹事　　（SWG1 担当）漆嶌　賢二　　エントラストジャパン株式会社

幹事　　（SWG1 担当）佐藤　雅史　　セコム株式会社

幹事　　（SWG2 担当）後藤　　淳　　日本電気株式会社

幹事　　（SWG3 担当）溝上　卓也　　日立ｿﾌﾄｳｪｱｴﾝｼﾞﾆｱﾘﾝｸﾞ株式会社

編集メンバ　（上記リーダ以外）

メンバ名	団体名
石原　達也	東芝ソリューション株式会社
南　　能之	株式会社 PFU
斉藤　　聡	株式会社リコー
川﨑　慎吾	大日本印刷株式会社
西川　康男＊	ARMA 東京支部
末武　陽一＊	関電システムソリューションズ（株）

＊：オブザーバ

メンバ（上記以外）

メンバ名	団体名
出本　　浩	株式会社エヌ・ティ・ティ・データ
保倉　　豊	グローバルフレンドシップ株式会社
鈴木　俊宏	日本オラクル株式会社
大窪　伸幸	株式会社 PFU
伊藤　信治	株式会社日立製作所
時得　克司	富士ゼロックス株式会社
鬼塚　正徳	富士電機ホールディングス株式会社
田中　　学	三菱電機株式会社
井上　雄二	株式会社リコー
大友 洋一	東北電力株式会社
政本　廣志	日本電信電話株式会社

電子文書長期保存ハンドブック

平成１９年　３月　発行

発　行　　次世代電子商取引推進協議会

販　売　　財団法人　日本情報処理開発協会
電子商取引推進センター
東京都港区芝公園三丁目５番８号
機械振興会館３階
TEL：０３（３４３６）７５００

この資料は再生紙を使用しています。