INTERNATIONAL STANDARD ISO/IEC 24761:2009
TECHNICAL CORRIGENDUM 1

Published 2013-03-01

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION
INTERNATIONAL ELECTROTECHNICAL COMMISSION • МЕЖДУНАРОДНАЯ ЭЛЕКТРОТЕХНИЧЕСКАЯ КОМИССИЯ • COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

Information technology — Security techniques — Authentication context for biometrics

TECHNICAL CORRIGENDUM 1

Technologies de l'information — Techniques de sécurité — Contexte d'authentification biométrique

RECTIFICATIF TECHNIQUE 1

Technical Corrigendum 1 to ISO/IEC 24761:2009 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

Page 1, Clause 2

Add the following at the end:

RFC 5911, New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME, June 2010

Page 11, Clause 6

Replace the definition of **`EncapsulatedContentInfoACBio`** with:

```
EncapsulatedContentInfoACBio ::= SEQUENCE {
        eContentType CONTENT-TYPE.&id({ContentTypeACBioContentInfo}),
        eContent [0] EXPLICIT OCTET STRING
        ( CONTAINING CONTENT-TYPE.&Type
            ({ContentTypeACBioContentInfo}{@eContentType}))}
```

ICS 35.040 **Ref. No. ISO/IEC 24761:2009/Cor.1:2013(E)**

Published in Switzerland

Page 11, Clause 6

Replace the definition of **ACBioContentInformation** and **Version** with:

```
ACBioContentInformation ::= SEQUENCE {
        version Version DEFAULT v1,
        bpuInformation BPUInformation,
        controlValue OCTET STRING (SIZE(16)),
        biometricProcess BiometricProcess,
        brtCertificateInformation BRTCertificateInformation OPTIONAL}

Version ::= INTEGER { v1(1) } ( v1, ... )
```

Page 15, 6.1

Replace the definition of **BPUReportInformation** with:

```
BPUReportInformation ::= CHOICE {
        bpuReport [0] EXPLICIT BPUReport,
        bpuReportReferrer URI}
```

Page 15, 6.2

Replace the definitions of ***BiometricProcess*** and ***SubprocessIndexList*** with:

```
BiometricProcess ::= SEQUENCE {
        subprocessIndexList SubprocessIndexList,
        bpuInputExecutionInformationList BPUIOExecutionInformationList OPTIONAL,
        bpuOuputExecutionInformationList BPUIOExecutionInformationList }

SubprocessIndexList ::= SEQUENCE SIZE(1..MAX) OF SubprocessIndex
```

Page 16, 6.3

Replace the definitions of ***BRTCertificateList*** and ***BRTCertificateReferrerList*** with:

```
BRTCertificateList ::= SEQUENCE SIZE(1..MAX) OF BRTCertificate

BRTCertificateReferrerList ::= SEQUENCE SIZE(1..MAX) OF URI
```

Page 17, 7.2

Replace the definitions of ***BPUReportInformation***, ***ContentTypeBPUReport***, ***EncapsulatedContentInfoBPUReport***, and ***contentBPUReport*** respectively with:

```
BPUReportInformation ::= CHOICE {
        bpuReport [0] EXPLICIT BPUReport,
        bpuReportReferrer URI}

ContentTypeBPUReport CONTENT-TYPE ::= {contentBPUReport }

EncapsulatedContentInfoBPUReport ::= SEQUENCE {
        eContentType CONTENT-TYPE.&id({ContentTypeBPUReportContentInfo}),
        eContent [0] EXPLICIT OCTET STRING
                ( CONTAINING CONTENT-TYPE.&Type
            ({ContentTypeBPUReportContentInfo}{@eContentType}))}

contentBPUReport CONTENT-TYPE ::= {
        ContentBPUReport
        IDENTIFIED BY id-contnetBPUReport}
```

Page 17, 7.2

Replace the second paragraph with:

BPUReport is defined in a similar way to **ACBioInstance**. A BPU Report consists of two fields; the first field of fixed value of **id-contnetBPUReport** and the second of type **ContentBPUReport**, which is a data of parameterized **SIGNEDDATA** with encapsulated content of type **BPUReportContentInformation**, which consists of two components, **bpuFunctionReport** and **bpuSecurityReport**. The signature shall be generated using the private key of the product vendor of the BPU.

NOTE The functions of and data flow in a BPU in enrolment may be different from those in biometric verification. In such a case, two **BPUReport**s may be prepared, one for enrolment, another for biometric verification. Otherwise one **BPUReport** may be prepared for both enrolment and biometric verification. The latter case is noted in 7.2.1.

In ASN.1 notation, **BPUReport** is described as follows:

Page 18, 7.2.1

Replace the definitions of ***BPUFunctionReport***, ***BPUSubprocessInformationList***, and ***BPUIOStaticInformationList*** respectively with:

```
BPUFunctionReport ::= SEQUENCE {
       bpuSubprocessInformationList BPUSubprocessInformationList,
       bpuInputStaticInformationList BPUIOStaticInformationList OPTIONAL,
       bpuOutputStaticInformationList BPUIOStaticInformationList }

BPUSubprocessInformationList ::= SEQUENCE SIZE(1..MAX) OF BPUSubprocessInformation

BPUIOStaticInformationList ::= SEQUENCE SIZE(1..MAX) OF BPUIOStaticInformation
```

Page 18, 7.2.1

Replace the last paragraph with:

bpuInputStaticInformationList is a list of elements of type **BPUIOStaticInformation** as many as the number of the input data to the BPU. **bpuOutputStaticInformationList** is a list of elements of type **BPUIOStaticInformation** as many as the number of the output data from the BPU. The type **BPUIOStaticInformation** is defined in 7.2.1.2.
In enrolment, storage subprocess shall output the hash value of the input of biometric sample which is to be stored as the biometric reference template, and the hash value is to be set in the BRT certificate. Therefore **bpuOutputStaticInformationList** shall have such a member if it is an expression for a BPU with storage subprocess in enrolment.

NOTE When the function of and data flow in a BPU in enrolment are different from those in biometric verification, the number of the elements in **bpuSubprocessInformationList** may not be equal to the number of the subprocesses in the BPU. It may be the sum of the number of the subprocesses in enrolment and that in biometric verification. In this case, **bpuSubprocessInformationList** is divided into two groups, one for enrolment and another for biometric verification. **subprocessName** of **functionDefinition** in a member of a group of **bpuSubprocessInformationList** may have the same value as the value of **subprocessName** of **functionDefinition** in a member in the other group but the value of the field **subprocessIndex** shall be different from that of the corresponding member of the list. If the **bpuSubprocessInformaitonList** is expressed as above, so are **bpuInputStaticInformationList** and **bpuOutputStaticInformationList** expressed in a similar way: there may be two members in the list where the value of **subprocessIOIndex** of one member is different from that of the other while the values of **dataType** are the same.

Page 19, 7.2.1.1.1

Replace the definition of **FunctionDefinition** with:

```
FunctionDefinition ::= SEQUENCE {
```

```
        subprocessName SubprocessName,
        subprocessIndex SubprocessIndex,
        biometricType BiometricType OPTIONAL,
        biometricSubtype BiometricSubtype OPTIONAL,
        inputIndex1 IOIndex OPTIONAL,
        inputIndex2 IOIndex OPTIONAL,
        outputIndex IOIndex,
        functionDescription OCTET STRING (SIZE(1..MAX)) OPTIONAL}
```

Page 19, 7.2.1.1.1

Add the following description after the third paragraph:

A pair of components **biometricType** and **biometricSubtype** indicates the modality of biometric data processed in the subprocess. The types **BiometricType** and **BiometricSubType** are defined in ISO/IEC 19785-3. **biometricType** is mandatory if **subprocessName** does not take the value **comparison** or **decision**.

Page 20, 7.2.1.2

Replace the first paragraph, definition of **BPUIOStaticInformation,** and second paragraph with:

BPUIOStaticInformation is a data type which gives information about input/output to/from the BPU, and consists of two components; **dataType** and **subprocessIOIndex**.

```
BPUIOStaticInformation ::= SEQUENCE {
        dataType DataType,
        subprocessIOIndex IOIndex}
```

Page 20, 7.2.1.2

Replace the fourth paragraph with:

There shall be the component **purpose** if the first component **processedLevel** takes the value from **raw-data** to **processed-data**. There shall not be the component purpose if the **processedLevel** takes the value **comparison-score**, **comparison-decision**, or **hashed-data**.

Page 20, 7.2.1.2

Replace the definition of **ProcessedLevel** with:

```
ProcessedLevel ::= ENUMERATED {
        raw-data(1),
        intermediate-data(2),
        processed-data(3),
        comparison-score(4),
        comparison-result(5),
        hashed-data(6),
        ...}
```

Page 21, Clause 8

Replace the text with:

BRT certificate is a certificate to the biometric reference template issued by a certain BRT certification organization. It contains information about the biometric reference template stored in the BPU, such as the issuer and validity period, etc.
Type **BRTCertificate** is defined similarly to **BPUReport**. A BRT certificate consists of two fields; the first field of fixed value of **id-contentBRTCertificate** and the second of type **ContentBRTCertificate**, which is a data of parameterized **SIGNEDDATA** with encapsulated content of type **BRTCContentInformation**. The signature shall be generated using the private key of the BRT certification organization.
In ASN.1 notation, BRTCertificate is described as follows:

```
BRTCertificate ::= SEQUENCE {
        contentType CONTENT-TYPE.&id({ContentTypeBRTCertificate}),
        content [0] EXPLICIT
            CONTENT-TYPE.&Type({ContentTypeBRTCertificate}{@contentType})}

ContentTypeBRTCertificate CONTENT-TYPE ::= { contentBRTCertificate }

ContentBRTCertificate ::= SIGNEDDATA { EncapsulatedContentInfoBRTCertificate }

EncapsulatedContentInfoBRTCertificate ::= SEQUENCE {
        eContentType CONTENT-TYPE.&id({ContentTypeBRTCertificateContentInfo}),
        eContent [0] EXPLICIT OCTET STRING
        ( CONTAINING CONTENT-TYPE.&Type
            ({ContentTypeBRTCertificateContentInfo}{@eContentType})}

ContentTypeBRTCertificateContentInfo CONTENT-TYPE ::= { brtcContentInformation }
```

The following attributes bind the type **ContentBRTCertificate** to **id-contentBRTCertificate**, and the type **BRTCContentInformation** to **id-brtcContentInformation**.
The types **BRTCertificate** and **EncapsulatedContentInfoBRTCertificate** are constrained with object sets containing a single object of class **CONTENT-TYPE** defined as follows:

```
contentBRTCertificate CONTENT-TYPE ::= {
        ContentBRTCertificate
        IDENTIFIED BY id-contentBRTCertificate  }

id-contentBRTCertificate  OBJECT IDENTIFIER ::=
        {iso(1) standard(0) acbio(24761) contentType(2) brtCertificate(6)}

brtcContentInformation CONTENT-TYPE ::= {
        BRTCContentInformation
        IDENTIFIED BY id-brtcContentInformation }

id-brtcContentInformation OBJECT IDENTIFIER ::=
        {iso(1) standard(0) acbio(24761) contentType(2) brtcContent(7)}
```

Page 22, 8.1

Replace the definitions of **SBHForBRTC**, **BDBForBRTC**, **HashList**, **UserInformation**, **PKICertificateInformation**, and **SequenceOfACBioInstances** respectively with:

```
SBHForBRTC ::= SEQUENCE {
        version CBEFFVersion,
        brtcIndex BIRIndex,
        brtcValidityPeriod BDBValidityPeriod,
        biometricType BiometricType,
        biometricSubtype BiometricSubtype OPTIONAL,
        brtQuality Quality,
        bdbEncryptionOptions EncryptionOptions(FALSE),
        bdbIntegrityOptions IntegrityOptions(FALSE),
        bdbFormatForBRTC BDBFormat}

BDBForBRTC ::= SEQUENCE {
        version Version DEFAULT v0,
```

```
        issuerAndSerialNumberBRTC IssuerAndSerialNumber OPTIONAL,
        originalBDBHashList HashList,
        originalBIRReferrer URI OPTIONAL,
        originalBIRPatronFormat PatronFormat,
        originalBDBPosition INTEGER,
        userInformation UserInformation OPTIONAL,
        pkiCertificateInformation PKICertificateInformation OPTIONAL,
        enrolmentACBioInstances SequenceOfACBioInstances OPTIONAL}

HashList ::= SEQUENCE SIZE(1..MAX) OF Hash

UserInformation ::= SEQUENCE {
        userIdentifier OCTET STRING,
        userName Name OPTIONAL,
        userUniqueIdentifier UniqueIdentifier OPTIONAL}

PKICertificateInformation ::= SEQUENCE {

        pkiCertificateSerialNumber CertificateSerialNumber,
        pkiCertificateIssuerName Name OPTIONAL,
        pkiCertificateIssuerUniqueIdentifier UniqueIdentifier OPTIONAL}

SequenceOfACBioInstances ::= SEQUENCE SIZE(1..MAX) OF ACBioInstance
```

Page 22, 8.1

Add the following description after the sixth paragraph of the explanation of the validity period of the BRT certificate:

biometricType together with **biometricSubtype** shows the modality of the biometric reference.

Page 22, 8.1

Add the following description after the twelfth paragraph of the explanation of the version of the format **BDBForBRTC**:

Optional field **issuerAndSerialNumberBRTC** of type **IssuerAndSerialNumberBRTC** imported from RFC 3852 is a pair of information, the issuer of the BRT certificate and the unique serial number issued by the issuer. OCSP may be applied to check the validity of the BRT certificate.

Page 24, Annex A

Replace the ASN.1 module with:

```
AuthenticationContextForBiometrics {iso(1) standard(0) acbio(24761) module(1) acbio(2)
version1(1)}

DEFINITIONS AUTOMATIC TAGS ::= BEGIN
IMPORTS

-- ASN.1 Module AlgorithmInformation in RFC 5912
        AlgorithmIdentifier
        FROM AlgorithmInformation-2009 {iso(1) identified-organization(3) dod(6)
internet(1) security(5)
            mechanisms(5) pkix(7) id-mod(0) id-mod-algorithmInformation-02(58)}

-- RFC 5280  revised as RFC 5912

        Certificate, CertificateList, CertificateSerialNumber, Name, UniqueIdentifier
        FROM PKIX1Explicit-2009 { iso(1) identified-organization(3) dod(6) internet(1)
            security(5) mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51) }
```

```
-- RFC 5755  revised as RFC 5912
        AttributeCertificate
        FROM PKIXAttributeCertificate-2009 { iso(1) identified-organization(3) dod(6)
internet(1)
            security(5) mechanisms(5) pkix(7) id-mod(0) id-mod-attribute-cert-02(47) }

-- ISO/IEC 19785 Common Biometric Exchange Formats Framework
        BiometricType, BiometricSubtype, CBEFFVersion, BIRIndex,
        BDBValidityPeriod,Quality,EncryptionOptions, IntegrityOptions,
        BDBFormat, PatronFormat
        FROM CBEFF-DATA-ELEMENTS {iso standard 19785 modules(0)
            types-for-cbeff-data-elements(1) }

-- RFC 3852 Cryptographic Message Syntax revised as RFC 5911
        CMSVersion, DigestAlgorithmIdentifier,
        SignerInfos, OriginatorInfo, RecipientInfos,
        MessageAuthenticationCodeAlgorithm, IssuerAndSerialNumber,
        AuthAttributes, MessageAuthenticationCode, UnauthAttributes,
        CONTENT-TYPE
        FROM CryptographicMessageSyntax2009{
            iso(1) member-body(2) us(840) rsadsi(113549)
            pkcs(1) pkcs-9(9) smime(16) modules(0) cms-2004-02(41) };

ACBioInstance ::= SEQUENCE {
        contentType CONTENT-TYPE.&id({ContentTypeACBio}),
        content [0] EXPLICIT CONTENT-TYPE.&Type
            ({ContentTypeACBio}{@contentType})}

ContentTypeACBio CONTENT-TYPE ::= {signedDataACBio |
                        authenticatedDataACBio}

SignedDataACBio ::= SIGNEDDATA { EncapsulatedContentInfoACBio }

AuthenticatedDataACBio ::= AUTHENTICATEDDATA { EncapsulatedContentInfoACBio }

EncapsulatedContentInfoACBio ::= SEQUENCE {
        eContentType CONTENT-TYPE.&id({ContentTypeACBioContentInfo}),
        eContent [0] EXPLICIT OCTET STRING
        ( CONTAINING CONTENT-TYPE.&Type
            ({ContentTypeACBioContentInfo}{@eContentType}))}

ContentTypeACBioContentInfo CONTENT-TYPE ::= {acbioContentInformation}

ACBioContentInformation ::= SEQUENCE {
        version Version DEFAULT v1,
        bpuInformation BPUInformation,
        controlValue OCTET STRING (SIZE(16)),
        biometricProcess BiometricProcess,
        brtCertificateInformation BRTCertificateInformation OPTIONAL}

Version ::= INTEGER { v1(1) } ( v1, ... )
```

```
BPUInformation ::= SEQUENCE {
        bpuCertificateReferrerInformation BPUCertificateReferrerInformation
            OPTIONAL,
        bpuReportInformation BPUReportInformation}

BPUCertificateReferrerInformation ::= SEQUENCE {
        bpuCertificateReferrer URI,
        crlsReferrer URI OPTIONAL}

URI ::= VisibleString (SIZE(1..MAX))

BPUReportInformation ::= CHOICE {
        bpuReport [0] EXPLICIT BPUReport,
        bpuReportReferrer URI}

BPUReport ::= SEQUENCE {
        contentType CONTENT-TYPE.&id({ContentTypeBPUReport}),
        content [0] EXPLICIT CONTENT-TYPE.&Type
            ({ContentTypeBPUReport}{@contentType})}

ContentTypeBPUReport CONTENT-TYPE ::= {contentBPUReport }

ContentBPUReport ::= SIGNEDDATA { EncapsulatedContentInfoBPUReport }

EncapsulatedContentInfoBPUReport ::= SEQUENCE {
        eContentType CONTENT-TYPE.&id({ContentTypeBPUReportContentInfo}),
        eContent [0] EXPLICIT OCTET STRING
                ( CONTAINING CONTENT-TYPE.&Type
            ({ContentTypeBPUReportContentInfo}{@eContentType}))}

ContentTypeBPUReportContentInfo CONTENT-TYPE ::= { bpuReportContentInformation }

BPUReportContentInformation ::= SEQUENCE {
        bpuFunctionReport BPUFunctionReport,
        bpuSecurityReport BPUSecurityReport}

BPUFunctionReport ::= SEQUENCE {
        bpuSubprocessInformationList BPUSubprocessInformationList,
        bpuInputStaticInformationList BPUIOStaticInformationList OPTIONAL,
        bpuOutputStaticInformationList BPUIOStaticInformationList }

BPUSubprocessInformationList ::= SEQUENCE SIZE(1..MAX) OF BPUSubprocessInformation

BPUSubprocessInformation ::= SEQUENCE {
        functionDefinition FunctionDefinition,
        qualityEvaluation QualityEvaluation OPTIONAL}

FunctionDefinition ::= SEQUENCE {
        subprocessName SubprocessName,
        subprocessIndex SubprocessIndex,
        biometricType BiometricType OPTIONAL,
        biometricSubtype BiometricSubtype OPTIONAL,
        inputIndex1 IOIndex OPTIONAL,
        inputIndex2 IOIndex OPTIONAL,
        outputIndex IOIndex,
        functionDescription OCTET STRING (SIZE(1..MAX)) OPTIONAL}

SubprocessName ::= ENUMERATED {
        data-capture(1),
        intermediate-signal-processing(2),
        final-signal-processing(3),
        storage(4),
        comparison(5),
        decision(6),
        sample-fusion(7),
        feature-fusion(8),
        score-fusion(9),
        decision-fusion(10),
        ...}

SubprocessIndex ::= INTEGER (0..65535)

IOIndex ::= INTEGER (0..65535)
```

```
QualityEvaluation ::= SEQUENCE {
       biometricProcessQualityInformation BiometricProcessQualityInformation  OPTIONAL,
       qualityEvaluationExtensionInformation QualityEvaluationExtensionInformation
OPTIONAL}

BiometricProcessQualityInformation ::= CHOICE {
       biometricProcessQuality BiometricProcessQuality,
       biometricProcessQualityReferrer URI}

QualityEvaluationExtensionInformation ::= CHOICE {

qualityEvaluationExtension QualityEvaluationExtension,
       qualityEvaluationExtensionReferrer URI}

BiometricProcessQuality ::= OCTET STRING (SIZE(1..MAX))

QualityEvaluationExtension ::= OCTET STRING (SIZE(1..MAX)) -- For extension

BPUIOStaticInformationList ::= SEQUENCE SIZE(1..MAX) OF BPUIOStaticInformation

BPUIOStaticInformation ::= SEQUENCE {
       dataType DataType,
       subprocessIOIndex IOIndex}

DataType ::= SEQUENCE {
       processedLevel ProcessedLevel,
       purpose Purpose OPTIONAL}

ProcessedLevel ::= ENUMERATED {
       raw-data(1),
       intermediate-data(2),
       processed-data(3),
       comparison-score(4),
       comparison-result(5),
       hashed-data(6),
       ...}

Purpose ::= ENUMERATED {
       reference(1),
       sample(2)}

BPUSecurityReport ::= SEQUENCE {
       cryptoModuleSecurityInformation CryptoModuleSecurityInformation
           OPTIONAL,
       biometricProcessSecurityInformation BiometricProcessSecurityInformation
           OPTIONAL,
       securityEvaluationExtensionInformation
           SecurityEvaluationExtensionInformation OPTIONAL}

CryptoModuleSecurityInformation ::= CHOICE {
       cryptoModuleSecurity CryptoModuleSecurity,
       cryptoModuleSecurityReferrer URI}

BiometricProcessSecurityInformation ::= CHOICE {
       biometricProcessSecurity BiometricProcessSecurity,
       biometricProcessSecurityReferrer URI}

SecurityEvaluationExtensionInformation ::= CHOICE {
       securityEvaluationExtension SecurityEvaluationExtension,
       securityEvaluationExtensionReferrer URI}

CryptoModuleSecurity ::= OCTET STRING (SIZE(1..MAX))

BiometricProcessSecurity ::= OCTET STRING (SIZE(1..MAX))

SecurityEvaluationExtension ::= OCTET STRING (SIZE(1..MAX)) -- For extension

BiometricProcess ::= SEQUENCE {
       subprocessIndexList SubprocessIndexList,
       bpuInputExecutionInformationList BPUIOExecutionInformationList OPTIONAL,
       bpuOuputExecutionInformationList BPUIOExecutionInformationList }

SubprocessIndexList ::= SEQUENCE SIZE(1..MAX) OF SubprocessIndex
```

```
BPUIOExecutionInformationList ::= SEQUENCE SIZE(1..MAX) OF BPUIOExecutionInformation

BPUIOExecutionInformation ::= SEQUENCE {
        dataType DataType,
        bpuIOIndex IOIndex,
        subprocessIOIndex IOIndex,
        hash Hash}

Hash ::= SEQUENCE {
        algorithmIdentifier AlgorithmIdentifier,
        hashValue OCTET STRING}

BRTCertificateInformation ::= CHOICE {
        brtCertificateList BRTCertificateList,
        brtCertificateReferrerList BRTCertificateReferrerList}

BRTCertificateList ::= SEQUENCE SIZE(1..MAX) OF BRTCertificate

BRTCertificateReferrerList ::= SEQUENCE SIZE(1..MAX) OF URI

BRTCertificate ::= SEQUENCE {
        contentType CONTENT-TYPE.&id({ContentTypeBRTCertificate}),
        content [0] EXPLICIT
            CONTENT-TYPE.&Type({ContentTypeBRTCertificate}{@contentType})}

ContentTypeBRTCertificate CONTENT-TYPE ::= { contentBRTCertificate }

ContentBRTCertificate ::= SIGNEDDATA { EncapsulatedContentInfoBRTCertificate }

EncapsulatedContentInfoBRTCertificate ::= SEQUENCE {
        eContentType CONTENT-TYPE.&id({ContentTypeBRTCertificateContentInfo}),
        eContent [0] EXPLICIT OCTET STRING
        ( CONTAINING CONTENT-TYPE.&Type
            ({ContentTypeBRTCertificateContentInfo}{@eContentType})}

ContentTypeBRTCertificateContentInfo CONTENT-TYPE ::= { brtcContentInformation }

BRTCContentInformation ::= SEQUENCE {
        sbhForBRTC SBHForBRTC,
        bdbForBRTC BDBForBRTC}

SBHForBRTC ::= SEQUENCE {
        version CBEFFVersion,
        brtcIndex BIRIndex,
        brtcValidityPeriod BDBValidityPeriod,
        biometricType BiometricType,
        biometricSubtype BiometricSubtype OPTIONAL,
        brtQuality Quality,
        bdbEncryptionOptions EncryptionOptions(FALSE),
        bdbIntegrityOptions IntegrityOptions(FALSE),
        bdbFormatForBRTC BDBFormat}

BDBForBRTC ::= SEQUENCE {
        version Version DEFAULT v1,
        issuerAndSerialNumberBRTC IssuerAndSerialNumber OPTIONAL,
        originalBDBHashList HashList,
        originalBIRReferrer URI OPTIONAL,
        originalBIRPatronFormat PatronFormat,
        originalBDBPosition INTEGER,
        userInformation UserInformation OPTIONAL,
        pkiCertificateInformation PKICertificateInformation OPTIONAL,
        enrolmentACBioInstances SequenceOfACBioInstances OPTIONAL}

HashList ::= SEQUENCE SIZE(1..MAX) OF Hash

UserInformation ::= SEQUENCE {
        userIdentifier OCTET STRING,
        userName Name OPTIONAL,
        userUniqueIdentifier UniqueIdentifier OPTIONAL}
```

```
PKICertificateInformation ::= SEQUENCE {

        pkiCertificateSerialNumber CertificateSerialNumber,
        pkiCertificateIssuerName Name OPTIONAL,
        pkiCertificateIssuerUniqueIdentifier UniqueIdentifier OPTIONAL}

SequenceOfACBioInstances ::= SEQUENCE SIZE(1..MAX) OF ACBioInstance

-- Useful definitions

SIGNEDDATA { EncapsulatedContentInfo } ::= SEQUENCE {
        version CMSVersion,
        digestAlgorithms SET OF DigestAlgorithmIdentifier,
        encapContentInfo EncapsulatedContentInfo,
        certificates [0] IMPLICIT CertificateSet OPTIONAL,
        crls [1] IMPLICIT RevocationInfoChoices OPTIONAL,
        signerInfos SignerInfos}

AUTHENTICATEDDATA { EncapsulatedContentInfo } ::= SEQUENCE {
        version CMSVersion,
        originatorInfo [0] IMPLICIT OriginatorInfo OPTIONAL,
        recipientInfos RecipientInfos,
        macAlgorithm MessageAuthenticationCodeAlgorithm,
        digestAlgorithm [1] DigestAlgorithmIdentifier OPTIONAL,
        encapContentInfo EncapsulatedContentInfo,
        authAttrs [2] IMPLICIT AuthAttributes OPTIONAL,
        mac MessageAuthenticationCode,
        unauthAttrs [3] IMPLICIT UnauthAttributes OPTIONAL}

CertificateSet ::= SET OF CertificateChoices

CertificateChoices ::= CHOICE {
        certificate Certificate,
        v2AttrCert [2] IMPLICIT AttributeCertificateV2,
        other [3] IMPLICIT OtherCertificateFormat}

AttributeCertificateV2 ::= AttributeCertificate

OtherCertificateFormat ::= SEQUENCE {
        otherFormat OTHERCERTIFICATE.&id({OtherCertificate}),
        otherCert OTHERCERTIFICATE.&Type({OtherCertificate}{@otherFormat})}

OTHERCERTIFICATE ::= TYPE-IDENTIFIER

OtherCertificate OTHERCERTIFICATE ::= {...}

RevocationInfoChoices ::= SET OF RevocationInfoChoice

RevocationInfoChoice ::= CHOICE {
        crl CertificateList,
        other [1] IMPLICIT OtherRevocationInfoFormat }

OtherRevocationInfoFormat ::= SEQUENCE {

otherRevInfoFormat OTHERREVOCATION.&id({OtherRevocation}),

otherRevInfo OTHERREVOCATION.&Type({OtherRevocation}{@otherRevInfoFormat}) }

OTHERREVOCATION ::= TYPE-IDENTIFIER

OtherRevocation OTHERREVOCATION ::= {...}

-- contentType object identifiers

id-signedDataACBio OBJECT IDENTIFIER ::=
        {iso(1) standard(0) acbio(24761) contentType(2) signedDataACBio(1)}

id-authenticatedDataACBio OBJECT IDENTIFIER ::=
        {iso(1) standard(0) acbio(24761) contentType(2) authenticatedDataACBio(2)}

id-acbioContentInformation OBJECT IDENTIFIER ::=
        {iso(1) standard(0) acbio(24761) contentType(2) acbioContent(3)}

id-contnetBPUReport OBJECT IDENTIFIER ::=
        {iso(1) standard(0) acbio(24761) contentType(2) bpuReport(4)}
```

```
id-bpuReportContentInformation OBJECT IDENTIFIER ::=
        {iso(1) standard(0) acbio(24761) contentType(2) bpuReportContent(5)}

id-contentBRTCertificate  OBJECT IDENTIFIER ::=
        {iso(1) standard(0) acbio(24761) contentType(2) brtCertificate(6)}

id-brtcContentInformation OBJECT IDENTIFIER ::=
        {iso(1) standard(0) acbio(24761) contentType(2) brtcContent(7)}

-- ContentType objects

signedDataACBio CONTENT-TYPE ::= {
        SignedDataACBio
        IDENTIFIED BY id-signedDataACBio }

authenticatedDataACBio CONTENT-TYPE ::= {
        AuthenticatedDataACBio
        IDENTIFIED BY id-authenticatedDataACBio  }

acbioContentInformation CONTENT-TYPE ::= {
        ACBioContentInformation
        IDENTIFIED BY id-acbioContentInformation }

contentBPUReport CONTENT-TYPE ::= {
        ContentBPUReport
        IDENTIFIED BY id-contnetBPUReport}

bpuReportContentInformation CONTENT-TYPE ::= {
        BPUReportContentInformation
        IDENTIFIED BY id-bpuReportContentInformation}

contentBRTCertificate CONTENT-TYPE ::= {
        ContentBRTCertificate
        IDENTIFIED BY id-contentBRTCertificate   }

brtcContentInformation CONTENT-TYPE ::= {
        BRTCContentInformation
        IDENTIFIED BY id-brtcContentInformation  }

END -- AuthenticationContextForBiometrics
```