INTERNATIONAL STANDARD

ISO/IEC 24761

First edition
2009-05-15

Information technology — Security techniques — Authentication context for biometrics

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

Reference number
ISO/IEC 24761:2009(E)

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

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

Introduction

A biometric verification process executed at a remote site is exposed to many risks: falsified reference templates, forged raw data, unreliable biometric devices, etc. How can the validator check whether a biometric verification process carried out in a remote site is trustworthy? This International Standard gives a mechanism to cope with this problem.

In general, reliability of the result of a biometric verification process is dependent both on the security level of the process executed and on the functional performance level of the biometric devices used. If devices offering a better functional performance level are used, the result will be more reliable. If the devices are not secure or the process has been executed in an unsecure environment, then the result will not be reliable.

In the Internet environment, the validator of a biometric verification process usually does not directly know about the biometric devices used or about the process(es) executed at a remote site. Obtaining trusted information, such as the functional performance level of the biometric devices used, the security level of the remote system, and also knowing that the processes in the system were executed securely, the validator can make a better decision on how much trust can be placed on the result of the biometric verification.

This International Standard provides a solution to the above problem by sending information about the devices used and the processes executed at the remote site to the validator.

In general, the biometric enrolment process consists of the following subprocesses: data capture, intermediate signal processing, final signal processing (or feature extraction), and storage. (This is true in general, but there are many variants possible.)

In general, the biometric verification process consists of data capture, intermediate signal processing, final signal processing, retrieval from storage, comparison, and decision. (This is true in general, but there are many variants possible.)

Usually, subprocesses are executed in one or more biometric processing units (BPUs), each of which has its own uniform security level. Several subprocesses are involved in the biometric verification process, but the security of the retrieval subprocess from storage is also dependent on the subprocesses involved in the biometric enrolment process.

This International Standard is designed to be applied to this model of biometric verification processes, which is an extension of the biometric system model defined in ISO 19092, but is also applicable to other biometric verification process models.

This International Standard defines a data format for security data generated by BPUs, such as a sensor, smartcard, or comparison device, to provide certified information about the BPU to help the validator to determine the reliability of the result of the biometric verification process.

This International Standard is based on the Public Key Infrastructure (PKI) technology and PKIX (see ISO/IEC 9594-8 | ITU-T Recommendation X.509 and RFC 3852). This International Standard uses a digital signature as the base for trust and non-repudiation. This International Standard requires input and output information to be hashed, and subsequently digitally signed with other data such as a challenge from the validator, and the evaluation result of the BPU actions.

This International Standard recognizes that privacy requirements concerned with the storage of biometric elements have to respond to and comply with local laws and legislation on data privacy. ACBio ensures that the validator can validate the result of the biometric verification process without receiving private data, such as the biometric sample acquired from the claimant or the biometric reference template used for comparison.

An ACBio instance is a report that is encoded using the XML Encoding Rules (XER) or the Basic Encoding Rules (BER) of ASN.1 [see ISO/IEC 8824 (All parts) | ITU-T Recommendations X.680-683 and ISO/IEC 8825-4 | ITU-T Recommendation X.693], commonly supported by cryptographic tool kit vendors. The syntax is algorithm independent and supports provision of data integrity and data origin authentication. The cryptographic algorithms specified by ISO/IEC JTC 1/SC 27 are recommended, though any algorithm appropriate for use by a given community may be used.

This International Standard uses BPU certificates, issued by a BPU certification organization (a trusted third party that issues certificates concerning the security of the BPU) and biometric reference template (BRT) certificates, issued by a BRT certification organization that issues certificates concerned with the production and retention of a biometric reference template in a database or on a smart-card.

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of a patent concerning the ACBio instance given in Clauses 5, 6, 7, and 8.

ISO and IEC take no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured the ISO and IEC that he/she is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with ISO and IEC. Information may be obtained from:

Toshiba Corporation, Toshiba Solutions Corporation,
1-1, Shibaura 1-chome, Minato-ku,
Tokyo 105-8001, Japan

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Information technology — Security techniques — Authentication context for biometrics

1 Scope

This International Standard defines the structure and the data elements of Authentication Context for Biometrics (ACBio), which is used for checking the validity of the result of a biometric verification process executed at a remote site. This International Standard allows any ACBio instance to accompany any data item that is involved in any biometric process related to verification and enrolment. The specification of ACBio is applicable not only to single modal biometric verification but also to multimodal fusion.

This International Standard specifies the cryptographic syntax of an ACBio instance. The cryptographic syntax of an ACBio instance is based on an abstract Cryptographic Message Syntax (CMS) schema whose concrete values can be represented using either a compact binary encoding or a human-readable XML encoding.

This International Standard does not define protocols to be used between entities such as BPUs, claimant, and validator. Its concern is entirely with the content and encoding of the ACBio instances for the various processing activities.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 8824 (all parts) | ITU-T Recommendations X.680-683, *Information technology — Abstract Syntax Notation One (ASN.1)*

ISO/IEC 8825-4 | ITU-T Recommendation X.693, *Information technology — ASN.1 encoding rules: XML Encoding Rules (XER)*

ISO/IEC 9594-2 | ITU-T Recommendation X.501, *Information technology — Open Systems Interconnection — The Directory: Models*

ISO/IEC 9594-8 | ITU-T Recommendation X.509, *Information technology — Open Systems Interconnection — The Directory: Public-key and attribute certificate frameworks*

ISO/IEC 19785-1, *Information technology — Common Biometric Exchange Formats Framework — Part 1: Data element specification*

ISO/IEC 19785-3, *Information technology — Common Biometric Exchange Formats Framework — Part 3: Patron format specifications*

RFC 3852, *Cryptographic Message Syntaxt (CMS), July 2004*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1
ACBio instance
report generated by a biometric processing unit (BPU) compliant to this International Standard to show the validity of the result of one or more subprocesses executed in the BPU

3.2
authentication context for biometrics
ACBio
International Standard that specifies the form and encoding of ACBio instances

3.3
biometric
pertaining to the field of biometrics

3.4
biometric processing unit
BPU
entity that executes one or more subprocesses that perform a biometric verification at a uniform level of security

NOTE A sensor, a smart card, and a comparison device are examples of BPUs.

3.5
biometric processing unit certificate
BPU certificate
X.509 certificate that is issued to a BPU by a certification organization which enables the validator to determine the reliability of the BPU

3.6
biometric processing unit certification organization
BPU certification organization
organization which issues BPU certificates

3.7
biometric processing unit function report
BPU function report
report on the function of the BPU, which contains the evaluation results on each function in the BPU

3.8
biometric processing unit IO Index
BPU IO Index
integer assigned to each biometric data stream between BPUs by the subject, such as software, which utilizes the function of the BPU so that the validator can reconstruct the data flow among BPUs

3.9
biometric processing unit report
BPU report
report on a BPU, which consists of a BPU function report and a BPU security report

3.10
biometric processing unit security report
BPU security report
report on the security level of a BPU, which contains an evaluation result of the security level of the BPU

3.11
biometric sample
information obtained from a biometric sensor, either directly or after further processing

NOTE See also **raw biometric sample** (3.33), **intermediate biometric sample** (3.26), and **processed biometric sample** (3.31).

3.12
biometric reference template
biometric sample or combination of biometric samples that has been stored as a reference for future comparison

NOTE See also **raw biometric reference template** (3.34), **intermediate biometric reference template** (3.27), and **processed biometric reference template** (3.32).

3.13
biometric reference template certificate
BRT certificate
certificate that is issued to a biometric reference template by a BRT certification organization and enables the validator to determine the authenticity of the biometric reference template

3.14
biometric reference template certification organization
BRT certification organization
organization which issues BRT certificates

3.15
biometric verification
application that returns true or false for a claim about the similarity of one or more biometric reference templates and one or more recognition biometric samples by making one or more comparisons

3.16
biometrics
automated recognition of individuals based on their behavioural and biological characteristics

3.17
claimant
⟨biometric verification⟩ individual who is seeking, and is the object of, biometric verification

3.18
comparison
estimation, calculation or measurement of similarity or dissimilarity between one or more recognition biometric samples and one or more biometric reference templates

3.19
comparison decision
determination of whether the recognition biometric sample(s) and biometric reference template(s) have the same biometric source, based on comparison scores, decision policies (including threshold values), and possibly other inputs to the comparison decision

3.20
comparison score
numerical value (or set of values) resulting from a comparison

3.21
control value
random number provided by a validator that is a means by which the validator can check whether an ACBio instance is generated at the validator's request or not

3.22
enrol
collect one or more biometric samples from an individual, and subsequently construct one or more biometric reference templates which can then be used to verify or determine the individual's identity

3.23
enrolment
process of collecting one or more biometric samples from an individual, and the subsequent construction of a biometric reference template which can then be used to verify or determine the individual's identity

3.24
enrolment organization
organization which handles enrolment and creates and stores biometric reference templates

3.25
evaluation organization
organization which evaluates biometric processing unit function or security

3.26
intermediate biometric sample
biometric sample obtained by processing a raw biometric sample, intended for further processing

3.27
intermediate biometric reference template
intermediate biometric sample or combination of intermediate biometric samples used as a biometric reference template

3.28
match
decision that the recognition biometric sample(s) and the biometric reference template are from the same individual

3.29
non-match
decision that the recognition biometric sample(s) and the biometric reference template are not from the same individual

3.30
on-card matching
performing comparison and decision making on an integrated circuit card where the biometric reference template is retained on-card in order to enhance security and privacy

NOTE The term "matching" is deprecated and replaced with the term "comparison" in ISO/IEC JTC 1/SC 37 work. But the term "on card matching" is a term heavily used in ISO/IEC JTC 1/SC 17 work. So this term is used in this International Standard.

3.31
processed biometric sample
biometric sample suitable for comparison

3.32
processed biometric reference template
processed biometric sample or combination of processed biometric samples used as a biometric reference template

3.33
raw biometric sample
biometric sample obtained directly from a biometric sensor

3.34
raw biometric reference template
raw biometric sample or combination of raw biometric samples used as a biometric reference template

3.35
subprocess
part of a biometric verification or enrolment process usually performing data capture, intermediate signal processing, final signal processing, storage, comparison, or decision

3.36
subprocess index
integer uniquely assigned to each subprocess within a biometric processing unit (BPU) by the organization providing the BPU

3.37
subprocess IO index
unique integer assigned to each data stream between subprocesses in a biometric processing unit (BPU) so that the validator can reconstruct the data flow between subprocesses in the BPU

3.38
validator
⟨biometric verification⟩ entity which makes a decision on whether the result of a biometric verification process is acceptable or not, based on the policy of the corresponding application, using one or more comparison decisions and possibly other information, supported by ACBio instances

4 Abbreviated terms

ACBio	Authentication Context for Biometrics
ASN.1	Abstract Syntax Notation One as defined in ISO/IEC 8824
BER	Basic Encoding Rules (of ASN.1)
BIR	Biometric Information Record
BPU	Biometric Processing Unit
BRT certificate	Biometric Reference Template certificate
CBEFF	Common Biometric Exchange Formats Framework as defined in ISO/IEC 19785-1
CMS	Cryptographic Message Syntax as defined in RFC 3852
IO	Input/Output
OCM	On-Card Matching
STOC	STore On Card
URI	Universal Resource Identifier
XML	eXtensible Markup Language

5 Model and framework of ACBio

5.1 Biometric enrolment and verification process model and Biometric Processing Unit (BPU)

ACBio's design is based on the following biometric verification subprocesses:

a) data capture

This subprocess captures biometric information from a claimant and converts it to a raw biometric sample. The raw biometric sample is transmitted to the intermediate signal processing subprocess for further processing.

b) intermediate signal processing

This subprocess receives a raw biometric sample and transforms it into an intermediate biometric sample. The intermediate biometric sample is transmitted to another intermediate signal processing subprocess or to the final signal processing subprocess for further processing.

c) final signal processing

This subprocess receives an intermediate biometric sample and transforms it into a processed biometric sample. The processed biometric sample is transmitted either to the comparison subprocess (verification) or to the storage subprocess (enrolment).

d) storage

This subprocess stores one of three types of biometric reference template; raw biometric reference template ((1) in Figure 1 and Figure 2), intermediate biometric reference template ((2) in Figure 1 and Figure 2), or processed biometric reference template ((3) in Figure 1 and Figure 2). One of the three types of biometric reference template will be compared with a biometric sample for verification.

e) comparison

This subprocess receives a biometric sample, which is acquired originally from a claimant, and may or may not be further processed, and a biometric reference template. This subprocess compares the biometric sample and the processed biometric reference template, and calculates the similarity, which is called a comparison score. The comparison score is transmitted to the decision subprocess.

f) decision

This subprocess receives a comparison score from the comparison subprocess, evaluates the score under rules determined by the security policy in use, decides the validity of the claimant's identity, and outputs the comparison decision, match or non-match, which is sent to the validator.

The following Figure 1 shows three biometric enrolment process models where the storage subprocess stores a raw biometric sample, an intermediate biometric sample, and a processed biometric sample.

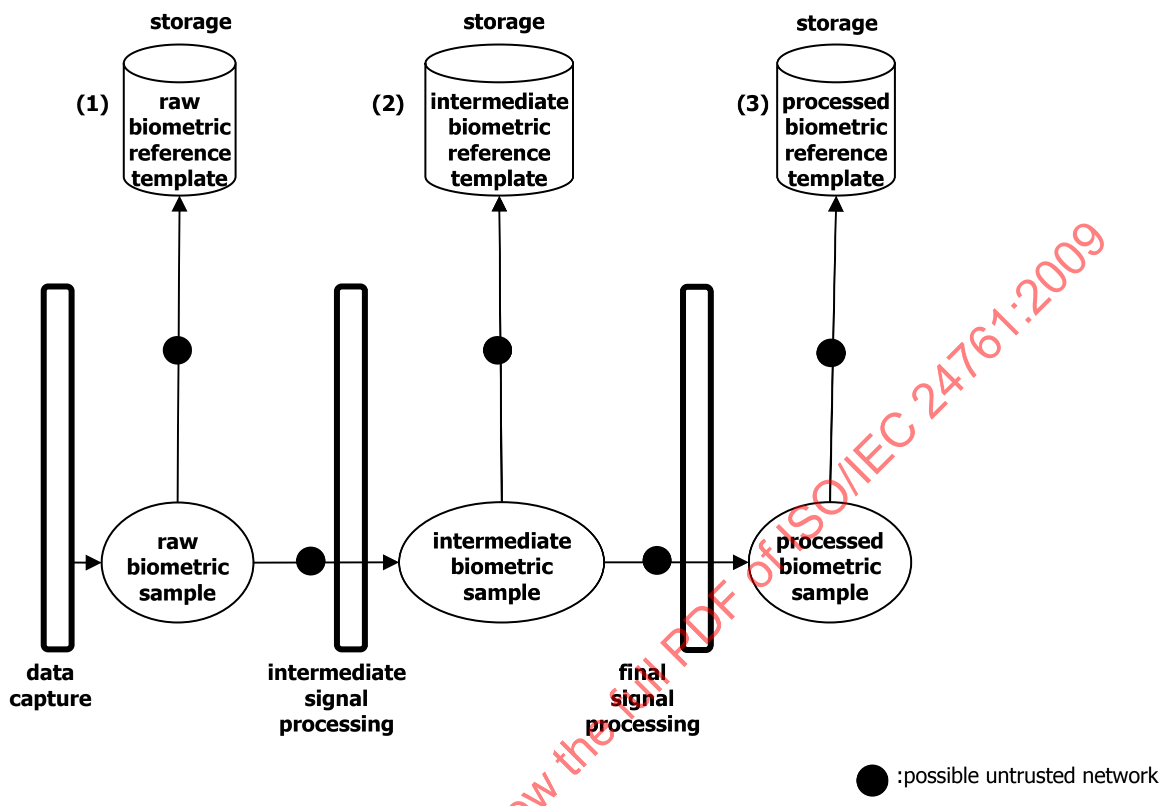

storage
storage
storage
(1)
raw biometric reference template
(2)
intermediate biometric reference template
(3)
processed biometric reference template
raw biometric sample
intermediate biometric sample
processed biometric sample
data capture
intermediate signal processing
final signal processing
● :possible untrusted network

Figure 1 — Biometric enrolment process model

The following Figure 2 shows three biometric verification process models where the storage subprocess stores a raw biometric reference template, an intermediate biometric reference template, and a processed biometric reference template.

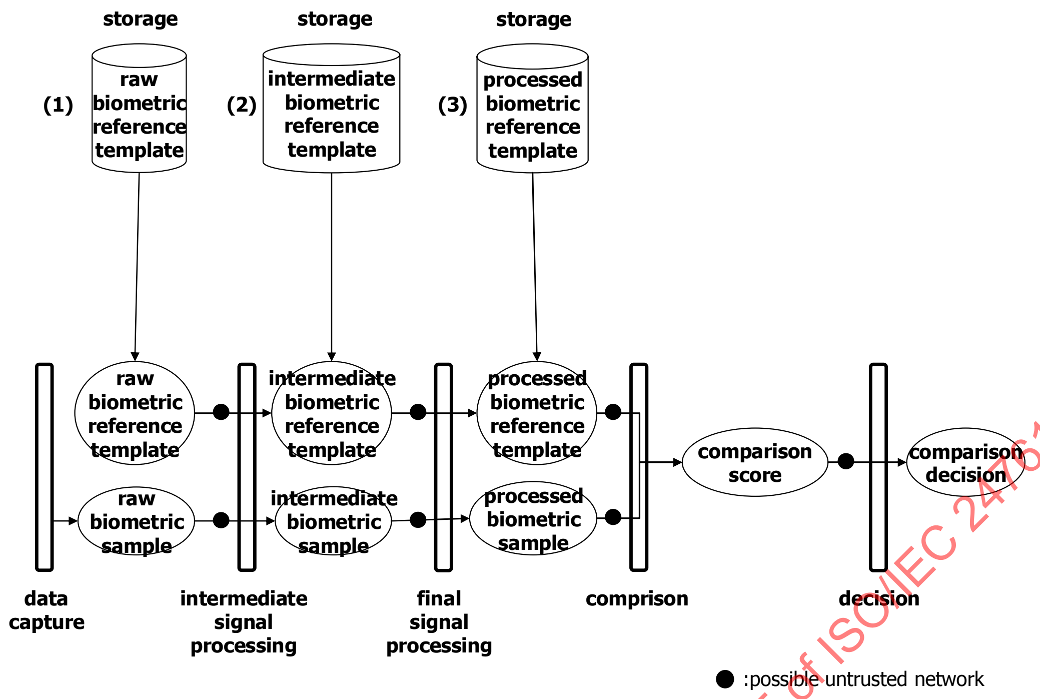

storage
storage
storage
(1)
raw biometric reference template
(2)
intermediate biometric reference template
(3)
processed biometric reference template
raw biometric reference template
intermediate biometric reference template
processed biometric reference template
raw biometric sample
intermediate biometric sample
processed biometric sample
comparison score
comparison decision
data capture
intermediate signal processing
final signal processing
comprison
decision
● :possible untrusted network

Figure 2 — Biometric verification process model

5.2 Framework for use of ACBio

ACBio gives information to the validator of a biometric verification process about the level of trust of the biometric verification process.

Subclauses 5.2.1 to 5.2.3 describe what shall be prepared in the production process of BPUs that result in a comparison and the enrolment process of BPUs that produce biometric references template, how ACBio instances are generated in these processes, and how biometric verification is validated.

5.2.1 Preparation for use of ACBio

To validate biometric verification processes with ACBio, preparation is necessary in addition to capture and storage (enrolment) of the biometric reference templates of claimants.

The series of steps in preparations for the use of ACBio is shown in Figure 3, separated into the production process, and the enrolment process, and the subsequent verification process.

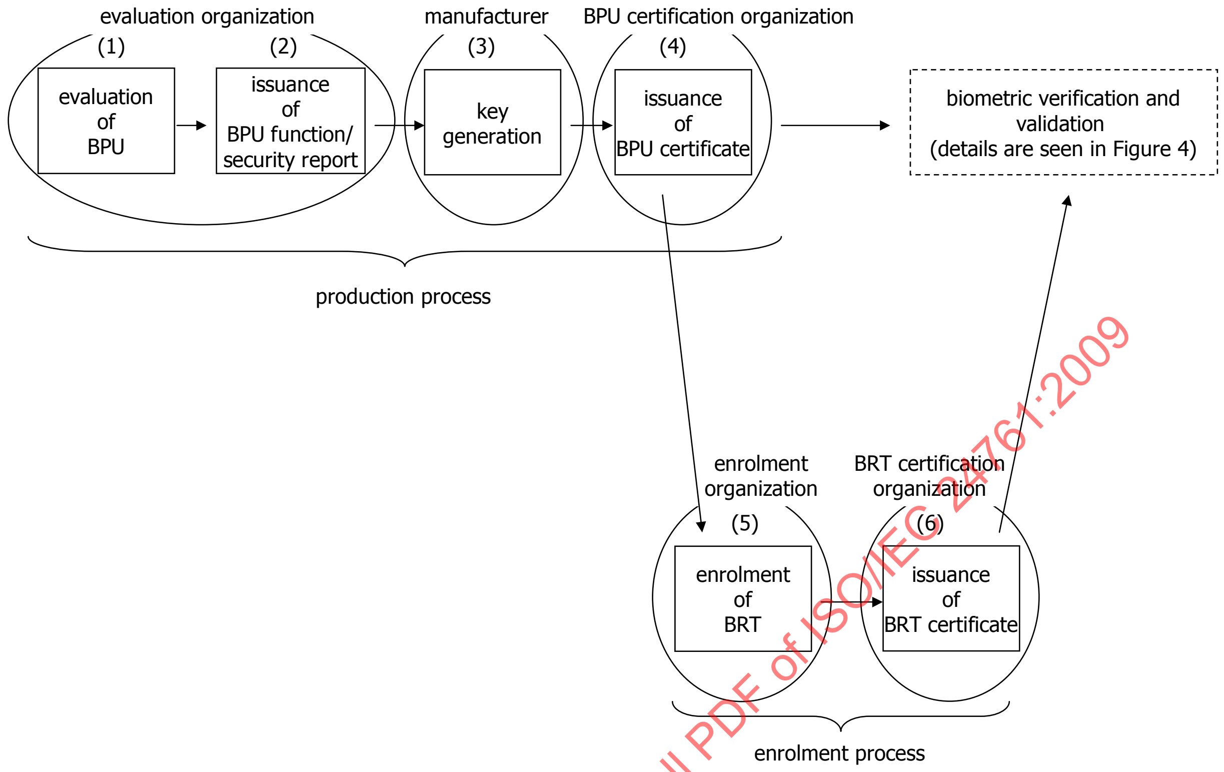

evaluation organization
(1)
(2)
evaluation of BPU
issuance of BPU function/ security report
manufacturer
(3)
key generation
BPU certification organization
(4)
issuance of BPU certificate
biometric verification and validation (details are seen in Figure 4)
production process
enrolment organization
(5)
enrolment of BRT
BRT certification organization
(6)
issuance of BRT certificate
enrolment process

Figure 3 — Preparation for use of ACBio and biometric verification execution

5.2.1.1 Preparation in the production process

The security level and functional performance level (quality) of each function in each BPU are evaluated by one or more evaluation organizations which may include the manufacturer of the product (software or hardware that forms the BPU ((1) in Figure 3).

After the evaluation, a BPU function report (See 7.2.1) and a BPU security report (See 7.2.2) are issued from the evaluation organization ((2) in Figure 2), which compose a BPU report.

Manufacturers of the products forming the BPU shall generate a key of symmetric cryptosystem or a key pair of asymmetric key cryptosystem depending on the BPU, to each BPU ((3) in Figure 3).

The key shall be certified and a BPU certificate (See 7.1) shall be issued by a BPU certification organization which may be the manufacturer of the product of the BPU ((4) in Figure 3).

The BPU report or a referrer to it, the BPU certificate or the referrer to it, and the key shall be stored in each BPU before shipping of the product of the BPU. Each BPU shall have the means to generate a digital signature or a message authentication code so that the validator can validate the integrity of the ACBio instance.

5.2.1.2 Preparation in the enrolment process

For biometric verification, a biometric reference template shall be created and enrolled in advance with an enrolment organization ((5) in Figure 3).

To use ACBio for validation of a biometric verification process, a certificate of biometric reference template called a BRT certificate (see Clause 8) shall be issued by a BRT certification organization ((6) in Figure 3).

The BRT certificate or the referrer to it shall be stored in the BPU where the certified biometric reference template is stored.

During the enrolment process, ACBio instances should be generated to validate the reliability of the enrolment process. The validation with these ACBio instances may be done by the storage subprocess as well as the validator of the biometric verification process.

5.2.2 Biometric verification and ACBio

Figure 4 shows the detail of the biometric verification process and its validation process.

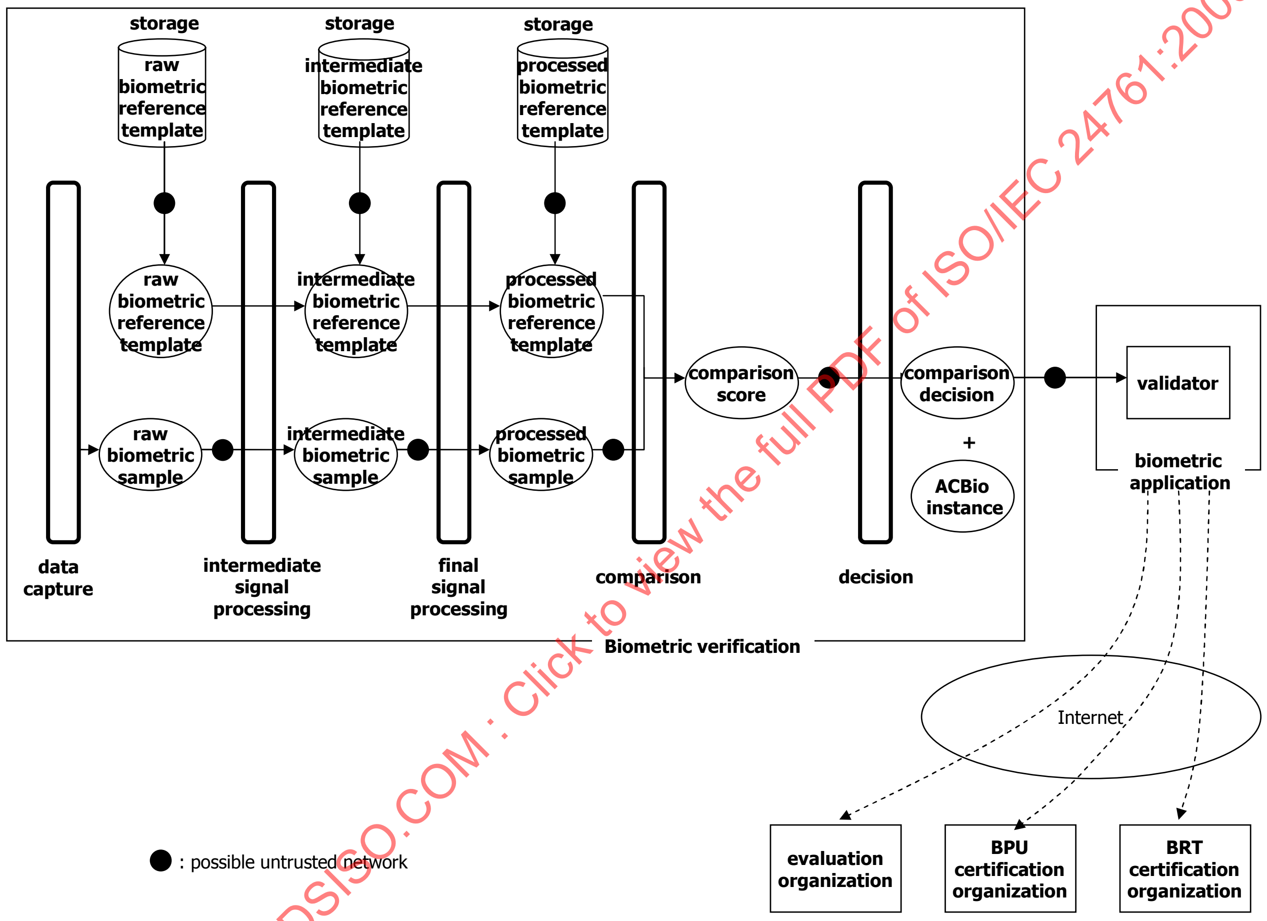

storage
raw biometric reference template
storage
intermediate biometric reference template
storage
processed biometric reference template
raw biometric reference template
intermediate biometric reference template
processed biometric reference template
raw biometric sample
intermediate biometric sample
processed biometric sample
comparison score
comparison decision
+
ACBio instance
validator
biometric application
data capture
intermediate signal processing
final signal processing
comparison
decision
Biometric verification
Internet
evaluation organization
BPU certification organization
BRT certification organization
● : possible untrusted network

Figure 4 — Biometric verification process and its validation

In a biometric verification process, each BPU shall pack its ACBio instance containing the BPU certificate information (the BPU certificate itself or the referrer to it), the BPU report information (the BPU report itself or the referrer to it), and the BRT certificate information (the BRT certificate itself or the referrer to it) if the BPU includes the storage subprocess into the data of type **ACBioContentInformation** (See 6). The data of type **ACBioContentInformation** shall contain a value from the validator, called the control value, and the hash value(s) of the input/output biometric data to/from the BPU, which enables the validator to validate the consistency of the transmission of biometric data between BPUs.

By adding the digital signature or the message authentication code to the data of type **ACBioContentInformation** with the key of the BPU, the ACBio instance is generated.

5.2.3 Validation of biometric verification process using ACBio

In this ACBio framework, the validator receives not only the comparison decision, the result of biometric verification, but also the ACBio instance(s) with which the validator can validate the result of the executed biometric verification.

The validator can validate the authenticity and integrity of an ACBio instance by verifying the digital signature or the message authentication code with the BPU certificate. The validator can obtain the security level and the functional performance level of the BPU by referring to the BPU report, and the authenticity of the biometric reference template used in the biometric verification process by referring to the BRT certificate. The validator can also validate the consistency of the communication between BPUs and between the validator and the biometric verification process by checking the biometric process block and the control value in the ACBio instance(s) respectively. With all of these, the validator can decide the level of trust for the result of the biometric verification.

If necessary, the validator can connect to relevant organizations such as the BPU certification organization, the evaluation organization, and the BRT certification organization, as shown in Figure 4.

6 ACBio instance

An ACBio instance is data of ASN.1 type **ACBioInstance** which is specified in ASN.1 notation as follows:

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

The type **ACBioInstance** corresponds to the type **ContentInfo** of CMS. The latter is constrained by an extensible object set while the former is constrained by an object set containing only two objects **signedDataACBio** and **authenticatedDataACBio**. These two objects of class **CONTENT-TYPE** are defined as follows:

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

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

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

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

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

SignedDataACBio and **AuthenticatedDataACBio** are specified as follows:

```
SignedDataACBio ::= SIGNEDDATA { EncapsulatedContentInfoACBio }

AuthenticatedDataACBio ::= AUTHENTICATEDDATA { EncapsulatedContentInfoACBio }
```

The types **SignedDataACBio** and **AuthenticatedDataACBio** specified above replace the CMS types **SignedData** and **AuthenticatedData** together with the following 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}
```

The following types appeared in the specification of the above two types, **SIGNEDDATA** and **AUTHENTICATEDDATA**, are imported from RFC 3852: **CMSVersion**, **DigestAlgorithmIdentifier**, **SignerInfos**, **OriginatorInfo**, **RecipientInfos**, **MessageAuthenticationCodeAlgorithm**, **AuthAttributes**, **MessageAuthenticationCode**, **UnauthAttributes**.

Since the types **CertificateSet** and **RevocationInfoChoices** in RFC 3852 are not compliant to the current version of ASN.1 which was standardized in 2002 as ISO/IEC 8824 (all parts) | ITU-T Recommendations X.680-683, they can not be imported and redefined as follows;

CertificateSet, **RevocationInfoChoices** are defined as follows;

```
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 ::= {...}
```

The type **EncapsulatedContentInfo** is a parameter in the above definition and is not imported from CMS. In the definition of the type **SignedDataACBio** and **AuthenticatedDataACBio**, the following type replaces **EncapsulatedContentInfo**.

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

ContentTypeACBioContentInfo CONTENT-TYPE ::= {acbioContentInformation}
```

As in the above definition, the type **EncapsulatedContentInfoACBio** is constrained by an object set containing a single object **acbioContentInformation** of the class **CONTENT-TYPE**. This object is defined as follows:

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

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

Therefore an ACBio instance is a data of type **ACBioInstance**, essentially the same as the CMS type **ContentInfo**, with the content of type **SignedDataACBio** or **AuthenticatedDataACBio** on the content of type **ACBioContentInformation**.

Table 1 shows the structure of type **ACBioContentInformation**. The **ACBioContentInformation** consists of five fields, version, BPU information block, control value, biometric process block, and BRT certificate information. The first four fields are mandatory. An ACBio instance shall have the last field if and only if the BPU contains the storage subprocess. The signature of **SignedDataACBio** or the message authentication code of **AuthenticatedDataACBio** shall be generated with the (private) key of the BPU.

Version is the version of the format of **ACBioContentInformation**.

Table 1 — ACBio Content Information

ACBioContentInformation		
	Version	
	BPU Information Block	
		BPU Certificate Referrer Information
		BPU Report Information
	Control Value	
	Biometric Process Block	
		SubprocessIndex[1]
		. .
		SubprocessIndex[L]
		BPUIOExecutionInformation[1] (for input)
		. .
		BPUIOExecutionInformation[M] (for input)
		BPUIOExecutionInformation[1] (for output)
		. .
		BPUIOExecutionInformation[N] (for output)
	BRT Certificate Information	

In ASN.1 notation, the type **ACBioContentInformation** is specified as follows:

```
ACBioContentInformation ::= [14] IMPLICIT SEQUENCE {
        version Version DEFAULT v0,
        bpuInformation BPUInformation,
        controlValue OCTET STRING (SIZE(16)),
        biometricProcess BiometricProcess,
        brtCertificateInformation BRTCertificateInformation OPTIONAL}
```

```
Version ::= INTEGER { v0(0) } ( v0, ... )
```

The tag of **ACBioContentInformation** is chosen so that the digital signature or message authentication code may be calculated for a data of this type on an IC card using the commands specified in ISO/IEC 7816-4.

The type **BPUInformation** is defined in 6.1. The detail of each type in **BPUInformation** is defined in Clause 7.

A control value is an octet string of 16 byte length with which the validator can check to which validator's request the ACBio instance is generated. It shall be set to **controlValue** field to make it infeasible to replay a biometric verification process.

The type **BiometricProcess** is defined in 6.2. The type **BRTCertificateInformation** is defined in 6.3. The detail of the type **BRTCertificate**, which is used in **BRTCertificateInformation**, is defined in Clause 8.

6.1 BPU information block

BPU information block carries the static information of BPU, information which does not depend on each execution. This block is mandatory and consists of two components, BPU certificate referrer information and BPU report information. ASN.1 type **BPUInformation** is defined for this block of information:

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

BPU certificate referrer information of type **BPUCertificateReferrerInformation** defined as follows is the referrer information to X.509 certificate for the (public) key of the BPU. If the ACBio instance contains the BPU certificate in the field of **certificates** in case of **SignedDataACBio** is used or in the field of **certs** in **originatorInfo** in case of **AuthenticatedDataACBio** is used, BPU certificate referrer information can be omitted. BPU certificate is specified in 7.1.

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

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

BPU report information of type **BPUReportInformation** is the BPU report itself or the referrer to it. BPU report contains the information about functions implemented in the BPU and the information of the security level of the BPU. BPU report is defined in 7.2.

```
BPUReportInformation ::= CHOICE {
        bpuReport BPUReport,
        bpuReportReferrer URI}
```

6.2 Biometric process block

Biometric process block carries the runtime information of BPU, information which depends on each execution. This block consists of three components, **subprocessIndexList** which is the list of indexes of the subprocesses executed in the BPU, **bpuInputExecutionInformationList** which contains the information on the input data to the BPU, and **bpuOuputExecutionInformationList** which contains the information on the output data from the BPU. If the BPU sends/receives data to/from other BPUs, then the corresponding components in this block are mandatory.

The ASN.1 type **BiometricProcess** is defined as follows:

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

SubprocessIndexList ::= SEQUENCE OF SubprocessIndex

BPUIOExecutionInformationList ::= SEQUENCE OF BPUIOExecutionInformation
```

subprocessIndexList is a list of data of type **SubprocessIndex**. This type is also used for **subprocessIndex** in type **FunctionDefinition** (See 7.2.1.1.1) which describes the function of a subprocess in a BPU. An ACBio instance contains as many data of type **FunctionDefinition** as the number of subprocesses in the BPU. The **subprocessIndex** in the type **FunctionDefinition** which corresponds to the executed subprocess shall be set to the above **subprocessIndexList**.

bpuInputExecutionInformationList consists of the elements of type **BPUIOExecutionInformation** as many as the input data to the BPU.

bpuOuputExecutionInformationList consists of the elements of type **BPUIOExecutionInformation** as many as the output data from the BPU.

For example, in case of a BPU which contains only the storage subprocess such as a STOC card, there is no **bpuInputExecutionInformationList** but **bpuOuputExecutionInformationList** with one element corresponding to output of the biometric reference template from the storage subprocess.

The definition for type **BPUIOExecutionInformation** is given as follows:

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

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

BPUIOInformation consists of four components, **dataType**, **bpuIOIndex**, **subprocessIOIndex**, and **hash**.

dataType indicates the type of the input/output data to/from the BPU. The type **DataType** is defined in 7.2.1.2.

On execution, the application program, which utilizes the function of the BPU, shall uniquely assign an integer to each biometric data stream from/to BPUs. Such an integer given by the application program shall be set to **bpuIOIndex**. If another BPU generates an ACBio instance with the same integer in **bpuIOIndex** in the biometric process block, it means that there was a communication between these two BPUs. In this way, the validator can reconstruct the data flow among BPUs.

The **subprocessIOIndex** of the corresponding element of the **bpuInputStaticInformationList/bpuOutputStaticInformationList** in the BPU information block shall be set to **subprocessIOIndex** of **BPUIOExecutionInformation**. The combination of **bpuIOIndex** and **subprocessIOIndex** makes the connection between the data flow inside BPU and the data flow in the whole biometric verification process.

hash contains the hash value of the input/output data to/from the BPU and the identifier of the hash algorithm. The type **AlgorithmIdentifier** is imported from ISO/IEC 9594-2 | ITU-T Recommendation X.501.

6.3 BRT certificate information

BRTcertificateInformation contains a list of the BRT certificates or the list of the referrer to each of the BRT certificates, as the following ASN.1 notation. BRT certificate contains the information about the biometric reference template stored in the BPU. An ACBio instance shall contain the BRT certificate information if and only if the BPU contains the storage subprocess. A list of more than one element is used if multi-modal fusion biometric verification is used. BRT certificate is specified in Clause 8.

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

BRTCertificateList ::= SEQUENCE OF BRTCertificate


BRTCertificateReferrerList ::= SEQUENCE OF URI
```

7 Definition of components in BPUInformationBlock

7.1 BPU certificate

A BPU certificate is the X.509 certificate for the (public) key of BPU. The structure of BPU certificate is described in Table 2.

Table 2 — BPU certificate

field		content
tbsCertificate	version	as ordinary
	serialNumber	as ordinary
	signature	as ordinary
	validity	as ordinary
	issuer	a trusted third party or a public CA in the vendor which produces/sells the product of the BPU
	subject	identifier of the subject including the information such as the serial number of the product, the name of the product of the BPU, and the name of the vendor of the product
	subjectPublicKeyInfo	as ordinary
	issuerUniqueIdentifier	as ordinary
	subjectUniqueIdentifier	as ordinary
	extensions	
signatureAlgorithm		as ordinary
signatureValue		as ordinary

The basic part of BPU certificate consists of nine fields; **version**, **serialNumber**, **signature**, **validity**, **issuer**, **subject**, **subjectPublicKeyInfo**, **issuerUniqueIdentifier**, and **subjectUniqueIdentifier**, all of which are the subfields of the field **tbsCertificate** of the type **Certificate** for X.509 certificate which is defined in ISO/IEC 9594-8 | ITU-T Recommendation X.509. Here the field **issuer** is a trusted third party or a public CA in the vendor which produces/sells the product of the BPU. The field **subject** is the identifier whose description is subject to ISO/IEC 9594-2 | ITU-T Recommendation X.501 and shall include the serial number of product, the product name and version of the BPU, and the name of the product vendor. The serial number of product in the subject field shall be the leaf entry of the identifier. The product name and version shall be the entry next to the leaf. Other seven attributes in the basic field are used as ordinary.

The BPU certificate shall be stored in the **certificates** field of **SignedDataACBio** or **AuthenticatedDataACBio** type field of the ACBio instance, or the referrer to the BPU certificate shall be stored in **bpuCertificateReferrerInformation** (see 6.1).

7.2 BPUReportInformation

BPU report information contains information about function(s) implemented in the BPU and information on the security level of the BPU. Either the BPU report itself or the referrer information to it shall be set in **BPUReportInformation**. In ASN.1 notation, **BPUReportInformation** is described as follows:

```
BPUReportInformation ::= CHOICE {
        bpuReport BPUReport,
        bpuReportReferrer URI}
```

BPUReport is defined in a similar way to **ACBioInstance.** A BPU Report consists of two fields; the first field of fixed value of **id-bpuReport** 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. In ASN.1 notation, **BPUReport** is described as follows:

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

ContentTypeBPUReport CONTENT-TYPE ::= {bpuReport}

ContentBPUReport ::= SIGNEDDATA { EncapsulatedContentInfoBPUReport }

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

ContentTypeBPUReportContentInfo CONTENT-TYPE ::= { bpuReportContentInformation }

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

BPUFunctionReport and **BPUSecurityReport** are defined in 7.2.1 and 7.2.2.

The types **BPUReport** and **BPUReportContentInformation** are constrained with object sets containing a single object of class **CONTENT-TYPE**. These objects are defined as follows:

```
bpuReport CONTENT-TYPE ::= {
            BPUReport
            IDENTIFIED BY id-bpuReport      }

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

7.2.1 BPUFunctionReport

BPU function report contains information about function(s) implemented in the BPU and input/output data to/from the BPU. The information about function includes the definition and functional performance level (quality) of the function. In ASN.1 notation, **BPUFunctionReport** is described as follows:

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

BPUSubprocessInformationList ::= SEQUENCE OF BPUSubprocessInformation

BPUIOStaticInformationList ::= SEQUENCE OF BPUIOStaticInformation
```

bpuSubprocessInformationList is a list of elements of type **BPUSubprocessInformation** as many as the number of the subprocesses implemented in the BPU. The type **BPUSubprocessInformation** is defined in 7.2.1.1.

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.

7.2.1.1 BPUSubprocessInformation

BPUSubprocessInformation contains information about the function and evaluation result of the subprocess, of type **FunctionDefinition** and **QualityEvaluation** defined in 7.2.1.1.1 and 7.2.1.1.2 respectively.

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

7.2.1.1.1 FunctionDefinition

FunctionDefinition consists of six components; **subprocessName**, **subprocessIndex**, **inputIndex1**, **inputIndex2**, **outputIndex**, and **functionDescription**.

subprocessName is of type **SubprocessName** and takes a value which represents the name of the subprocess.

To each subprocess in the BPU, the vendor of the product of the BPU shall assign a unique integer. **subprocessIndex** is such an index given to the subprocess.

To each data stream which comes into or goes from any subprocess in the BPU, the vendor of the product of the BPU shall assign an integer. These integers shall be assigned uniquely within the BPU. If an input/output to/from a subprocess is given, then it is on one of the streams and is given the integer assigned to the data stream naturally. **inputIndex1**, **inputIndex2**, and **outputIndex** are given in this way. Any subprocess except data capture shall have inputIndex1. Comparison subprocess shall have both inputIndex1 and inputIndex2.

descriptionFunction is for supplementary description of the function of the subprocess.

The ASN.1 notation for type **FunctionDefinition** is given as follows:

```
FunctionDefinition ::= SEQUENCE {
        subprocessName SubprocessName,
        subprocessIndex SubprocessIndex,
        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)
```

7.2.1.1.2 QualityEvaluation

QualityEvaluation consists of **biometricProcessQualityInformation** and **qualityEvaluationExtensionInformation. biometricProcessQualityInformation** contains either an evaluation report on biometric performance test of the subprocess or the referrer to it. **qualityEvaluationExtensionInformation** is the field for future extension. It contains a report or the referrer to it. In ASN.1 notation, **QualityEvaluation** is described as follows:

```
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
```

NOTE **BiometricProcessQuality** and **QualityEvaluationExtension** are not specified in this International Standard. In SC 37/WG 5, some parts of ISO/IEC 19795 on biometric performance testing and reporting has been standardized and others are being standardized at the current moment. And machine readable format of biometric performance testing and reporting has begun its standardization as ISO/IEC 29120 in SC 37/WG 5. When it is standardized, it is recommended to use the machine readable format specified in ISO/IEC 29120 as **BiometricProcessQuality**. Before it is standardized, these fields are used with specification defined by vendor, user or their association.

7.2.1.2 BPUIOStaticInformation

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

```
BPUIOStaticInformation ::= SEQUENCE {
            biometricType BiometricType OPTIONAL,
            biometricSubtype BiometricSubtype OPTIONAL,
            dataType DataType,
            subprocessIOIndex IOIndex}
```

A pair of components **biometricType** and **biometricSubtype** indicates the modality of biometric data of input/output to/from the BPU. **biometricType** is mandatory if **processedLevel** of **dataType** field does not take either **comparison-score** or **comparison-decision**. The types **BiometricType** and **BiometricSubType** are defined in ISO/IEC 19785-3.

dataType is of type **DataType** which consists of two components, **processedLevel** and **purpose**. The former takes a value which corresponds to one of raw data, intermediate data, processed data, comparison score, or comparison decision. The latter takes a value which corresponds to biometric reference template or biometric sample.

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** or **comparison-decision**.

An input/output to/from a BPU is one of input/output to/from a subprocess in the BPU. **subprocessIOIndex** shall be the value of the corresponding **inputIndex1/inputIndex2/outputIndex** of a certain data of type **FunctionDefinition** in BPU subprocess information.

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

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

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

7.2.2 BPUSecurityReport

BPUSecurityReport consists of three components, **CryptoModuleSecurityInformation**, **BiometricProcessSecurityInformation**, and **SecurityEvaluationExtensionInformation**. Each type contains either an evaluation report or the referrer to it. In ASN.1 notation, **BPUSecurityReport** is described as follows:

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

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

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

NOTE **CryptoModuleSecurity** and **BiometricProcessSecurity** are used to indicate the security level of the module/process by reference to an appropriate indicator. The reference may be to a standardized or proprietary security level.

```
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
```

8 BRT certificate

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-brtCertificate** and the second of type **ContentBRTCert**, 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] CONTENT-TYPE.&Type({ContentTypeBRTCertificate}{@contentType})}

ContentTypeBRTCertificate CONTENT-TYPE ::= { brtCertificate }

ContentBRTCertificate ::= SIGNEDDATA { EncapsulatedContentInfoBRTCertificate }

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

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

The following attributes bind the type **BRTCertificate** to **id-brtCertificate**, 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:

```
brtCertificate CONTENT-TYPE ::= {
        BRTCertificate
        IDENTIFIED BY id-brtCertificate }

id-brtCertificate 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)}
```

8.1 BRTCContentInformation

BRTCContentInformation is expressed with CBEFF BIR (Biometric Information Record) which is specified in ISO/IEC 19785-1. **BRTCContentInformation** consists of two parts, **sbhForBRTC** and **bdbForBRTC**. To express the former, SBH (Standard Biometric Header) of CBEFF is applied. The latter is a newly defined BDB (Biometric Data Block) format for BRT certificate.

Type **BRTCContentInformation** is described as follows:

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

sbhForBRTC is of type **SBHForBRTC** and has seven elements; **version**, **brtcIndex**, **brtcValidityPeriod**, **brtQuality**, **bdbEncryptionOptions**, **bdbIntegrityOptions** , and **bdbFormatForBRC**. The type for each element is specified in ISO/IEC 19785-3.

version is used for specifying the version of format of **SHBForBRTC**.

brtcIndex indicates the index of the BRT certificate.

brtcValidityPeriod contains the validity period of the BRT certificate.

brtQuality contains the quality of the biometric reference template.

optionBDBEncryption and **optionBIRIntegrity** are encryption option and integrity option, and shall be set **FALSE**.

bdbFormatForBRTC indicates the format owner and format type of **BDBForBRTC**.

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

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

bdbForBRTC is of type **BDBForBRTC** and has eight elements; **version**, **originalBDBHashList**, **originalBIRReferrer**, **originalBIRpatronFormat**, **originalBDBPosition**, **userInformation**, **pkiCertificateInformation**, and **enrolmentACBioInstances**.

version is the version of the format **BDBForBRTC**.

originalBDBHashList is a list of **Hash**. **Hash** contains two fields, hash value and algorithm identifier of hash algorithm. The former is the hash value of the biometric reference template. If **originalBDBHashList** contains more than one element, they are of a single biometric reference template and of different hash algorithm.

originalBIRReferrer is the referrer to the original BIR.

originalBDBPosition indicates the position of the biometric reference template corresponding to this BRT certificate in the original BDB.

userInformation is an optional field of type **UserInformation** which contains identifier, name, and unique identifier of the person whose biometric reference template is the object of the BRT certificate.

pkiCertificateInformation is an optional field and contains information about X.509 public key certificate of the user, the serial number of the certificate, the name of issuer, and the unique identifier of the certificate. This field links BRT certificate to X.509 certificate.

enrolmentACBioInstances is an optional list of ACBio instances generated at the enrolment of the biometric reference template.

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

```
BDBForBRTC ::= SEQUENCE {
        version Version DEFAULT v0,
        originalBDBHashList HashList,
        originalBIRReferrer URI OPTIONAL,
        originalBIRPatronFormat PatronFormat,
        originalBDBPosition INTEGER,
        userInformation UserInformation OPTIONAL,
        pkiCertificateInformation PKICertificateInformation OPTIONAL,
        enrolmentACBioInstances SequenceOfACBioInstances OPTIONAL}

HashList ::= SEQUENCE OF Hash

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

PKICertificateInformation ::= SEQUENCE {
        pkiCertificateSerialNumber CertificateSerialNumber,
        pkiCertificateIssuerName Name,
        pkiCertificateIssuerUniqueIdentifier UniqueIdentifier OPTIONAL}

SequenceOfACBioInstances ::= SEQUENCE OF ACBioInstance
```

8.2 Format Owner and Format Type values

The Format Owner for the **BDBForBRTC** shall use the value 0102 hex (258 decimal), which is the registered Patron Format Identifier for ISO/IEC JTC1 SC27. The Format Type for the BRC shall use the value 0001 hex (1 decimal), which has been registered by SC27 as the value for the **BDBForBRTC**.

The resulting ASN.1 Object Identifier value for the **BDBForBRTC** is thus:

```
{iso registration-authority cbeff(19785) organization(0) iso-iec-jtc1-SC27 (258)
        bdbs(0) biometric-reference-template-certificate (1)}
```

Annex A
(normative)

ASN.1 module for ACBio

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

DEFINITIONS AUTOMATIC TAGS ::= BEGIN
IMPORTS

-- ISO/IEC 9594-8 Open Systems Interconnection --
-- The Directory: Authentication framework
        AlgorithmIdentifier, CertificateSerialNumber, Certificate
        FROM AuthenticationFramework {joint-iso-itu-t ds(5) module(1)
            authenticationFramework(7) 5}

-- AttributeCertificate
        AttributeCertificate
        FROM AttributeCertificate {
            joint-iso-itu-t ds(5) module(1) attributeCertificateDefinitions(32) 5}

-- CertificateExtensions
        CertificateList
        FROM CertificateExtensions {
            joint-iso-itu-t ds(5) module(1) certificateExtensions(26) 5}

-- ITU-T X.501 Open Systems Interconnection   The Directory: Models
        Name
        FROM InformationFramework {joint-iso-itu-t ds(5) module(1)
            informationFramework(1) 5}

        UniqueIdentifier
        FROM SelectedAttributeTypes {joint-iso-itu-t ds(5) module(1)
            selectedAttributeTypes(5) 5}

-- 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
        CMSVersion, DigestAlgorithmIdentifier,
        SignerInfos, OriginatorInfo, RecipientInfos,
        MessageAuthenticationCodeAlgorithm,
        AuthAttributes, MessageAuthenticationCode, UnauthAttributes,
        CONTENT-TYPE
        FROM CryptographicMessageSyntax2004 {
            iso(1) member-body(2) us(840) rsadsi(113549)
            pkcs(1) pkcs-9(9) smime(16) modules(0) cms-2004(24) };

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 CONTENT-TYPE.&Type
            ({ContentTypeACBioContentInfo}{@eContentType})}

ContentTypeACBioContentInfo CONTENT-TYPE ::= {acbioContentInformation}

ACBioContentInformation ::= [14] IMPLICIT SEQUENCE {
        version Version DEFAULT v0,
        bpuInformation BPUInformation,
        controlValue OCTET STRING (SIZE(16)),
        biometricProcess BiometricProcess,
        brtCertificateInformation BRTCertificateInformation OPTIONAL}

Version ::= INTEGER { v0(0) } ( v0, ... )

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

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

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

BPUReportInformation ::= CHOICE {
        bpuReport BPUReport,
        bpuReportReferrer URI}

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

ContentTypeBPUReport CONTENT-TYPE ::= {bpuReport}

ContentBPUReport ::= SIGNEDDATA { EncapsulatedContentInfoBPUReport }

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

ContentTypeBPUReportContentInfo CONTENT-TYPE ::= { bpuReportContentInformation }

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

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

BPUSubprocessInformationList ::= SEQUENCE OF BPUSubprocessInformation

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

FunctionDefinition ::= SEQUENCE {
        subprocessName SubprocessName,
        subprocessIndex SubprocessIndex,
        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 OF BPUIOStaticInformation

BPUIOStaticInformation ::= SEQUENCE {
            biometricType BiometricType OPTIONAL,
            biometricSubtype BiometricSubtype OPTIONAL,
            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-decision(5),
            ...}

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 OPTIONAL}

SubprocessIndexList ::= SEQUENCE OF SubprocessIndex

BPUIOExecutionInformationList ::= SEQUENCE 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 OF BRTCertificate

BRTCertificateReferrerList ::= SEQUENCE OF URI

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

ContentTypeBRTCertificate CONTENT-TYPE ::= { brtCertificate }

ContentBRTCertificate ::= SIGNEDDATA { EncapsulatedContentInfoBRTCertificate }

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

ContentTypeBRTCertificateContentInfo CONTENT-TYPE ::= { brtcContentInformation }

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

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

BDBForBRTC ::= SEQUENCE {
        version Version DEFAULT v0,
        originalBDBHashList HashList,
        originalBIRReferrer URI OPTIONAL,
        originalBIRPatronFormat PatronFormat,
        originalBDBPosition INTEGER,
        userInformation UserInformation OPTIONAL,
        pkiCertificateInformation PKICertificateInformation OPTIONAL,
        enrolmentACBioInstances SequenceOfACBioInstances OPTIONAL}

HashList ::= SEQUENCE OF Hash
```

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

PKICertificateInformation ::= SEQUENCE {
        pkiCertificateSerialNumber CertificateSerialNumber,
        pkiCertificateIssuerName Name,
        pkiCertificateIssuerUniqueIdentifier UniqueIdentifier OPTIONAL}

SequenceOfACBioInstances ::= SEQUENCE 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-bpuReport 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-brtCertificate 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     }

bpuReport CONTENT-TYPE ::= {
        BPUReport
        IDENTIFIED BY id-bpuReport      }

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

brtCertificate CONTENT-TYPE ::= {
        BRTCertificate
        IDENTIFIED BY id-brtCertificate  }

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

END -- AuthenticationContextForBiometrics
```

Annex B
(informative)

Implementation examples

B.1 Examples of the implementation for ACBio

In this International Standard, the protocol for ACBio is not specified. Here in this annex, two examples of implementation of ACBio are given including protocols; one for a case of STOC (STore On Card) model, the other for a case of OCM (On card matching) model.

B.1.1 An Example of the implementation for STOC Model

In this example, we assume that this STOC model of a biometric verification process consists of two BPUs, one is a biometric device which has the functions of data capture, intermediate signal processing, final signal processing, comparison and decision, and the other is a STOC card which stores the processed biometric reference template. This example is mainly focused on STOC card.

B.1.1.1 In evaluation process

Products of BPUs, i.e. STOC cards and biometric devices used in a biometric verification process, should be evaluated at a certain evaluation organization and issued their BPU reports.

B.1.1.2 In production process

Vendors of BPUs should index every subprocess and stream in accordance with the rule in 7.2.1. If the subprocesses and streams in the biometric device and those in the STOC card are indexed as in Figure B.1, then the BPUFunctionReport of the biometric device and that of the STOC card are as shown in Figure B.2. In Figure B.1, SIndex means subprocess index and IOIndex means subprocess IO index.

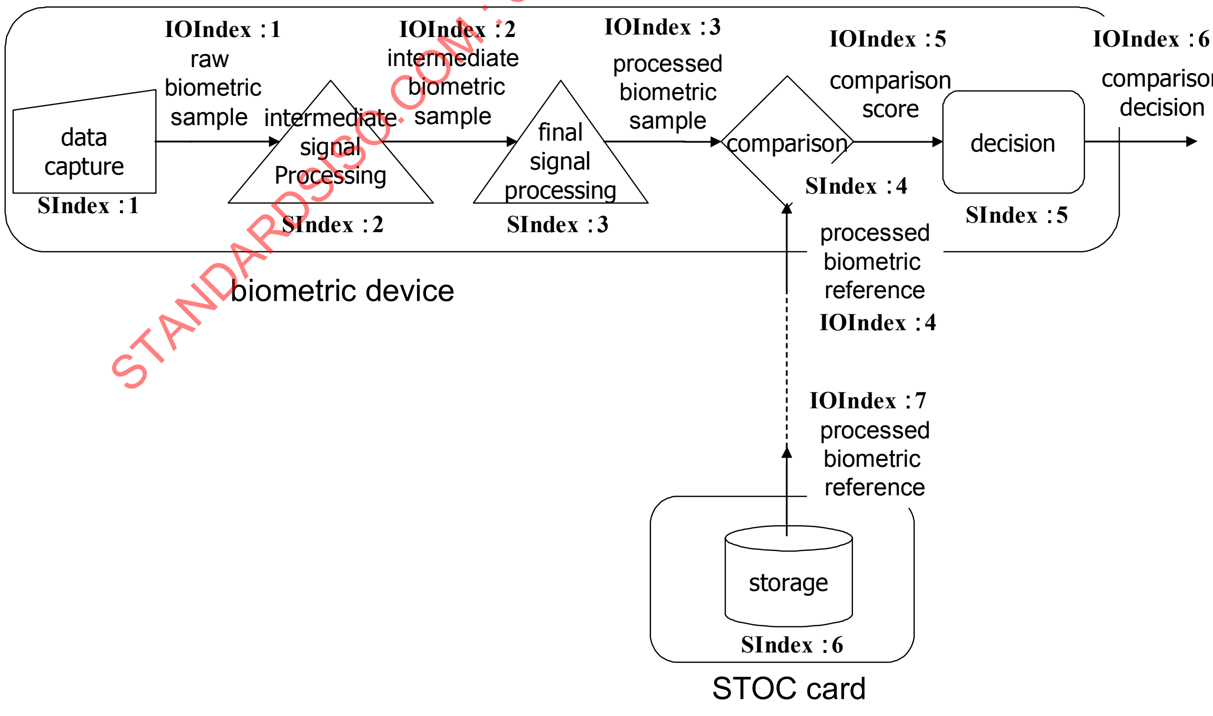


Figure B.1 — A biometric verification process of a STOC model and an example of indexing

- BPUFunctionReport
  - BPUSubprocessInformation
    - FunctionDefinition
      - data-capture (name of function)
      - 1 (subprocess index)
      - 1 (index of output)
      - DescriptionFunction
    - QualityEvaluation
    - FunctionDefinition
      - intermediate-signal-processing (name of function)
      - 2 (subprocess index)
      - 1 (index of input1)
      - 2 (index of output)
      - DescriptionFunction
    - QualityEvaluation
    - FunctionDefinition
      - final-signal-processing (name of function)
      - 3 (subprocess index)
      - 2 (index of input1)
      - 3 (index of output)
      - DescriptionFunction
    - QualityEvaluation
    - FunctionDefinition
      - comparison (name of function)
      - 4 (subprocess index)
      - 3 (index of input1)
      - 4 (index of input2)
      - 5 (index of output)
      - DescriptionFunction
    - QualityEvaluation
    - FunctionDefinition
      - decision (name of function)
      - 5 (subprocess index)
      - 5 (index of input1)
      - 6 (index of output)
      - DescriptionFunction
    - QualityEvaluation
  - BPUIOInformation (for input)
    - BiometricType
    - BiometricSubtype
    - TypeData
      - processed-data (processed level)
      - reference (purpose)
    - 4 (subprocess IO index)
  - BPUIOInformation (for output)
    - BiometricType
    - BiometricSubtype
    - TypeData
      - comparison-decision (processed level)
    - 6 (subprocess IO index)

BPUFunctionReport of the biometric device

- BPUFunctionReport
  - BPUSubprocessInformation
    - FunctionDefinition
      - storage (name of function)
      - 6 (subprocess index)
      - 7 (index of output)
      - DescriptionFunction
    - QualityEvaluation
  - BPUIOInformation (for output)
    - BiometricType
    - BiometricSubtype
    - TypeData
      - processed-data (processed level)
      - reference (purpose)
    - 7 (subprocess IO index)

BPUFunctionReport of the STOC card

Figure B.2 — Examples of BPUFunctionReports for a STOC model

In this case, a block of data of the type **ACBioContentInformation** should be stored in a STOC card in advance as in the Figure B.3. In Figure B.3, the data elements marked with * (shaded) are fixed data and set in the area of a STOC card in advance while those not marked with * (unshaded) are left blank to be filled later in the enrolment and execution processes.

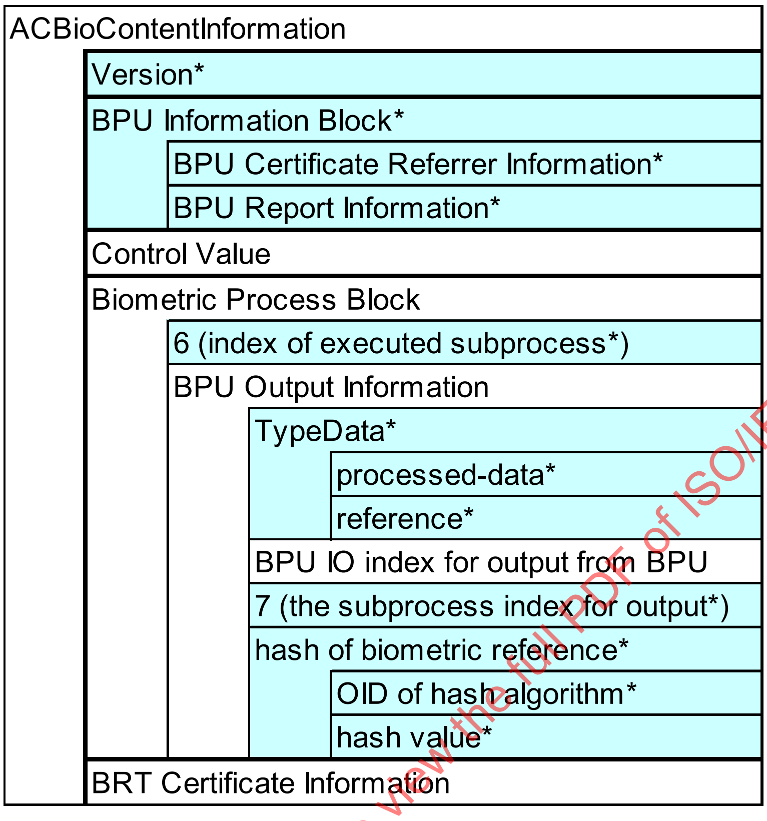


Figure B.3 — A block of data stored in a STOC card on production process

In Figure B.3, the fields not marked with * (unshaded) except BRT certificate Information have fixed length of their own. The length of the field for BRT certificate Information is not determined on the production process but determined in the enrolment process. It is noted that the area for **ACBioContentInformation** should have sufficient successive area for BRT certificate Information.

The following table is the BER-TLV format of **ACBioContentInformation** for a STOC card, which is equivalent to the data structure of Figure B.3.

Table B.1 — BER-TLV format of ACBioContentInformation for a STOC card

Tag	L	Value								
'AE'	Var.	ACBioContentInformation								
		Tag	L	Value						
		'02'	1	Version						
		'A0'	Var.	BPU Information Block						
				Tag	L	Value				
				'A0'	Var.	BPU Certificate Referrer Information				
				'A1'	Var.	BPU Report Information				
		'04'	16	Control Value						
		'A2'	Var.	Biometric Process Block						
				Tag	L	Value				
				'02'	1	index of executed subprocess1				
				'A1'	Var.	Output Information				
						Tag	L	Value		
						'A0'	6	Type Data		
								Tag	L	Value
								'0A'	1	processed-data
								'0A'	1	reference
						'02'	1	BPU IO index for output to BPU		
						'02'	1	subprocess index for output		
						'A1'	Var.	Hash information of output		
								'06'	Var.	OID of hash algorithm
								'04'	Var.	Hash value of output
		'A3'	Var.	BRT Certificate Information						

B.1.1.3 In enrolment process

Biometric reference template is stored to a STOC card in this process. A BRT certificate is issued to the biometric reference template, and the BRT certificate or the referrer to it is stored in **brtCertificateInformation** of **ACBioContentInformation**. The length of the field of the type **ACBioContentInformation** should be adjusted by adding the length of BRT certificate Information plus the length of Tag field and Length field.

B.1.1.4 In exectution process

On an execution of biometric verification, two inputs are given to a STOC card; the first is a control value from the validator, the second is the BPU IO index to the output from the STOC card. The STOC card should set the first to the “Control Value” ((1) PUT DATA in the Figure B.4), the second to the “BPU IO index for output from BPU” ((2) PUT DATA in the Figure B.4), respectively.

Subsequently the STOC card digitally signs the whole field of the type **ACBioContentInformation** ((3) PERFORM SECURITY OPERATION in Figure B.4) to get the ACBio instance and sends it as the output ((5) GET DATA in Figure B.4)) together with the processed biometric reference template ((4) GET DATA in Figure B.4)).

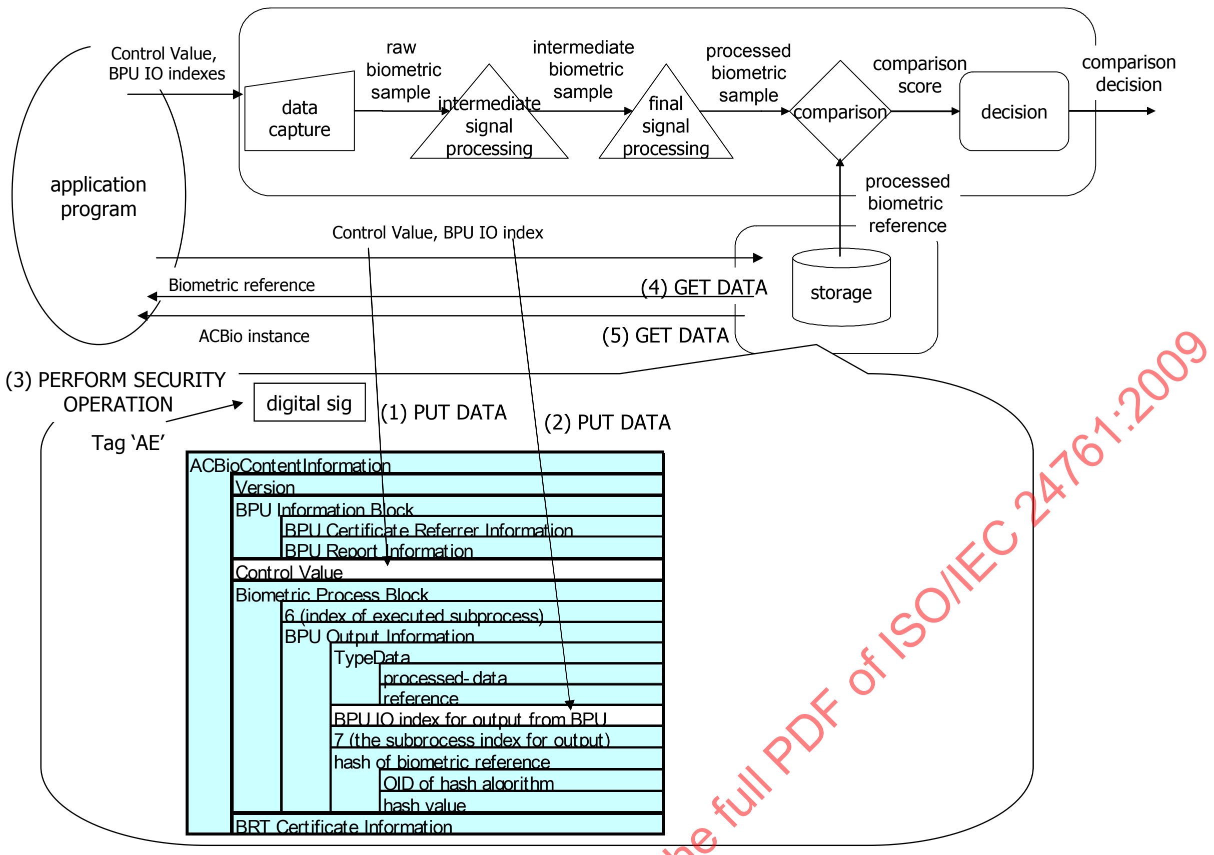


Figure B.4 — ACBio instance generation on a STOC card on an execution of a biometric verification

Figure B.5 describes a command sequence for ACBio instance generation on a STOC card where BIT is the abbreviation for Biometric Information Template, which is a term used in ISO/IEC 7816 series.

Command/Response	Meaning
PUT DATA <Control Value> → ← OK	Put control value as an input parameter
PUT DATA <BPU IO Index Output> → ← OK	Put BPU IO index for output as an input parameter
PERFORM SECURITY OPERATION <'AE', 'BC'> → ← OK	Calculate digital signature for ACBioContentInformation and put it to the digital signature field of SignedData structure of ACBio instance
GET DATA → ← BIT	Get processed biometric reference as the output
GET DATA → ← BIT	Get ACBio instance as the output

Figure B.5 — Command sequence for ACBio instance generation on a STOC card

B.1.1.5 An example of protocol

We assume the existence of a system of the validator side, and a system (device) of the claimant side which only has function to transmit/receive messages to/from the other three, as shown in Figure B.6.

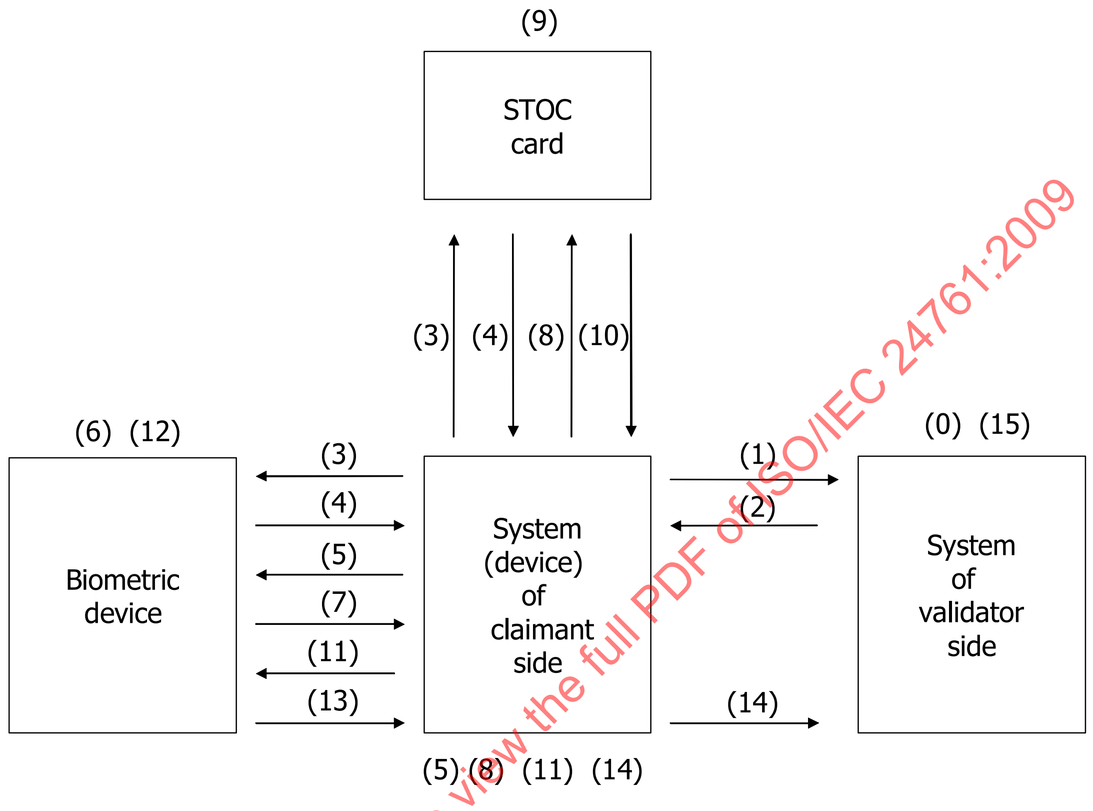


Figure B.6 — An example of protocol for a STOC model

An example of the protocol for this STOC model is as follows:

(0) Beforehand, the validator sets up the ACBio verification policy for each of the data elements of ACBio instances based on the policy of the corresponding application (see B.3 for an example of ACBio verification policy).

(1) The claimant requests biometric verification to the validator, via the system (device) of claimant side.

(2) The system of validator side sends the control value and the candidate list of hash algorithm and digital signature algorithm according to the ACBio verification policy, and requests the execution of biometric verification including the generation of ACBio instance(s) to the system (device) of claimant side.

(3) The system (device) of the claimant inquires the algorithms available to the biometric device and the STOC card.

(4) The biometric device and the STOC card return the algorithms available to the system (device) of claimant side.

(5) The system (device) of claimant side decides the hash algorithm and the signature algorithm, and sends the validator's control value, the hash algorithm, and the signature algorithm to the biometric device and requests execution of the data capture, the intermediate signal processing, and the final signal processing to the biometric device.

(6) The biometric device captures the biometric information from the claimant and generates the processed biometric sample through the data capture, intermediate signal processing, and final signal processing subprocesses.

(7) The biometric device sends the termination message of the final signal processing to the system (device) of the claimant side.

(8) The system (device) of the claimant side sends the validator's control value, the hash algorithm and the signature algorithm selected in step (5), and a BPU IO Index for the output data (processed biometric reference template) of the STOC card, to the STOC card. And also the system (device) of the claimant side requests generation of an ACBio instance, and transmission of a processed biometric reference template and the ACBio instance, to the STOC card.

(9) The STOC card generates an ACBio instance. See B.1.1.4 for the detail.

(10) The STOC card sends the processed biometric reference template and the ACBio instance to the system (device) of the claimant side.

(11) The system (device) of claimant side sends the processed biometric reference template received from the STOC card, the BPU IO index assigned to the processed biometric reference template, and the BPU IO Index assigned to the comparison decision of the biometric device, to the biometric device. And also, the system (device) of claimant side requests execution of the comparison and decision subprocesses to the biometric device.

(12) The biometric device receives the processed biometric reference template and executes the comparison and decision subprocesses. The biometric device also generates an ACBio instance by the following procedures;

a) Restore the BPU Information block and set it to **bpuInformation** of **ACBioContentInformation**.

b) Set the validator’s control value to **controlValue** of **ACBioContentInformation**.

c) Generate the biometric process block as follows;

c-1) Set the subprocess indexes to **subprocessIndexList**. The subprocess indexes are assigned by the product vendor of the biometric device to the each of the subprocesses corresponding to the function of data capture, intermediate signal processing, final signal processing, comparison, and decision.

c-2) To make **bpuInputExecutionInformationList** field, the following should be done. Set the value **processed-data** and the value **reference** respectively to **processedLevel** and **purpose** of **dataType**. Set the BPU IO index, assigned to the input of the biometric device, to **bpuIOIndex**. Set the subprocess IO index of the input data (processed biometric reference template) to the comparison subprocess, which was assigned by the product vendor of the biometric device, to **subprocessIOIndex**. Set the pair of the hash value of the processed biometric reference template received and the hash algorithm to **hash**.

c-3) To make **bpuOutputExecutionInformationList** field, the following should be done. Set the value **comparison-decision** to **processedLevel** of **dataType**. Set the BPU IO index, assigned to the output data of the biometric device, to **bpuIOIndex**. Set the subprocess IO index of the output data (comparison decision) of the decision subprocess, which was assigned by the product vendor of the biometric device, to **subprocessIOIndex**. Set the pair of the hash value of the comparison decision and the hash algorithm into **hash**.

d) Generate **SignedDataACBio** of the data which consists of the data generated by a), b) and c), using the digital signature algorithm selected in (5). If the BPU certificate is put in **certificates** of **SignedDataACBio**, the BPU certificate referrer information in the BPU information block can be omitted.

(13) The biometric device sends the output data of the decision subprocess and the ACBio instance to the system (device) of the claimant side.

(14) The system (device) of the claimant side sends the comparison decision (the output data of the decision subprocess) and the two ACBio instances to the system of the validator side.

(15) The system of the validator side receives the comparison decision and the two ACBio instances. The validator validates the results with the following procedures;

a) Validate the integrity of each ACBio instance by signature verification.

b) Validate the correspondence of the control value originally given by the validator and the control value of each ACBio instance.

c) Validate that the security level of two BPUs and the functional performance level of each subprocess executed in two BPUs satisfy the ACBio verification policy of the validator (see B.3 for the example of ACBio verification policy). The information about the security level of two BPUs and the functional performance level of each subprocess executed in the two BPUs is in the BPU report which is stored in or referred from the BPU information block of each ACBio instance.

d) Validate the validity of biometric reference template used. The information about the biometric reference template, such as the issuer, the validity period, etc., is in the BRT certificate, which is stored in or referred from the BRT certificate Information of ACBio instance generated by the STOC card.

e) Validate that the whole process of biometric verification executed at the claimant side by checking the executed subprocesses. Subprocess indexes corresponding to executed subprocesses are stored in **subprocessIndexList** in the biometric process blocks of the two ACBio instances, and the executed functions corresponding to those subprocess indexes can be identified from **subprocessName** of **functionDefinition** of the two BPU reports.

f) Validate the consistency of the input data and the output data transmitted between BPUs by comparing the contents of **dataType**, **bpuIOIndex**, and **hashValue** of **bpuOutputExecutionInformationList** in the biometric process block of the ACBio instance generated by the STOC card and the contents of **bpuIOIndex**, **dataType**, and **hashValue** of **bpuInputExecutionInformationList** in the biometric process block of the ACBio instance generated by the biometric device.

Figure B.7 illustrates the above validation process of the biometric verification process using ACBio instances. In Figure B.7, the indexes are given as in Figure B.1.

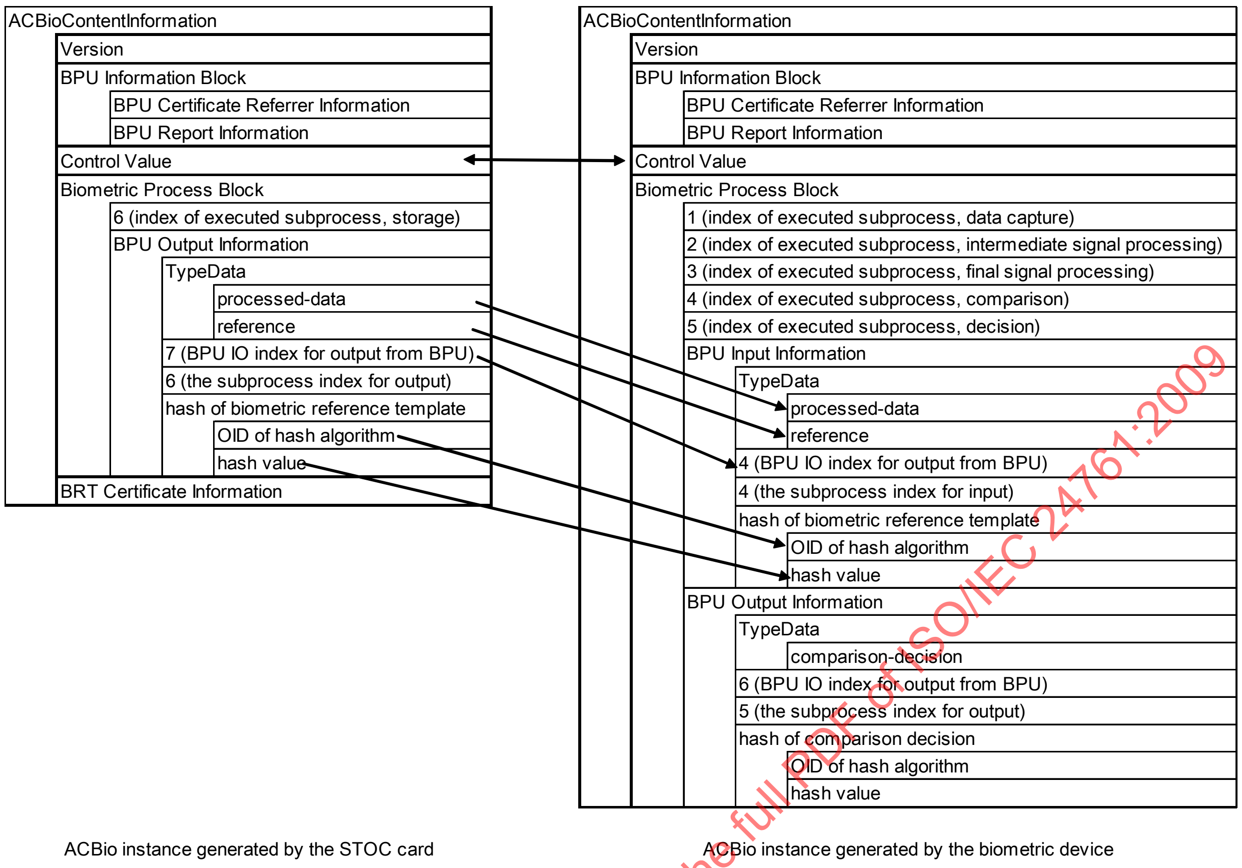

ACBioContentInformation
Version
BPU Information Block
BPU Certificate Referrer Information
BPU Report Information
Control Value
Biometric Process Block
6 (index of executed subprocess, storage)
BPU Output Information
TypeData
processed-data
reference
7 (BPU IO index for output from BPU)
6 (the subprocess index for output)
hash of biometric reference template
OID of hash algorithm
hash value
BRT Certificate Information
ACBioContentInformation
Version
BPU Information Block
BPU Certificate Referrer Information
BPU Report Information
Control Value
Biometric Process Block
1 (index of executed subprocess, data capture)
2 (index of executed subprocess, intermediate signal processing)
3 (index of executed subprocess, final signal processing)
4 (index of executed subprocess, comparison)
5 (index of executed subprocess, decision)
BPU Input Information
TypeData
processed-data
reference
4 (BPU IO index for output from BPU)
4 (the subprocess index for input)
hash of biometric reference template
OID of hash algorithm
hash value
BPU Output Information
TypeData
comparison-decision
6 (BPU IO index for output from BPU)
5 (the subprocess index for output)
hash of comparison decision
OID of hash algorithm
hash value
ACBio instance generated by the STOC card
ACBio instance generated by the biometric device

Figure B.7 — Validation of biometric verification using ACBio

B.1.2 An Example of the implementation for OCM Model

In this example, we assume that this OCM model of biometric verification process consists of two BPUs, one is a sensor device which has the functions of data capture, intermediate signal processing, and final signal processing, and the other is an OCM card which has the functions of storage, comparison, and decision. This example is mainly focused on OCM card.

B.1.2.1 In evaluation process

Products of BPUs, i.e. OCM cards and sensor devices used in a biometric verification process, should be evaluated at a certain evaluation organization and issued their BPU reports.

B.1.2.2 In production process

Vendors of BPUs should index every subprocess and stream in accordance with the rule in 7.2.1. If the subprocesses and streams in the sensor device and those in the OCM card are indexed as in Figure B.8, then the BPUFunctionReport of the biometric device and that of the OCM card are as shown in Figure B.9. In Figure B.8, SIndex means subprocess index and IOIndex means subprocess IO index.

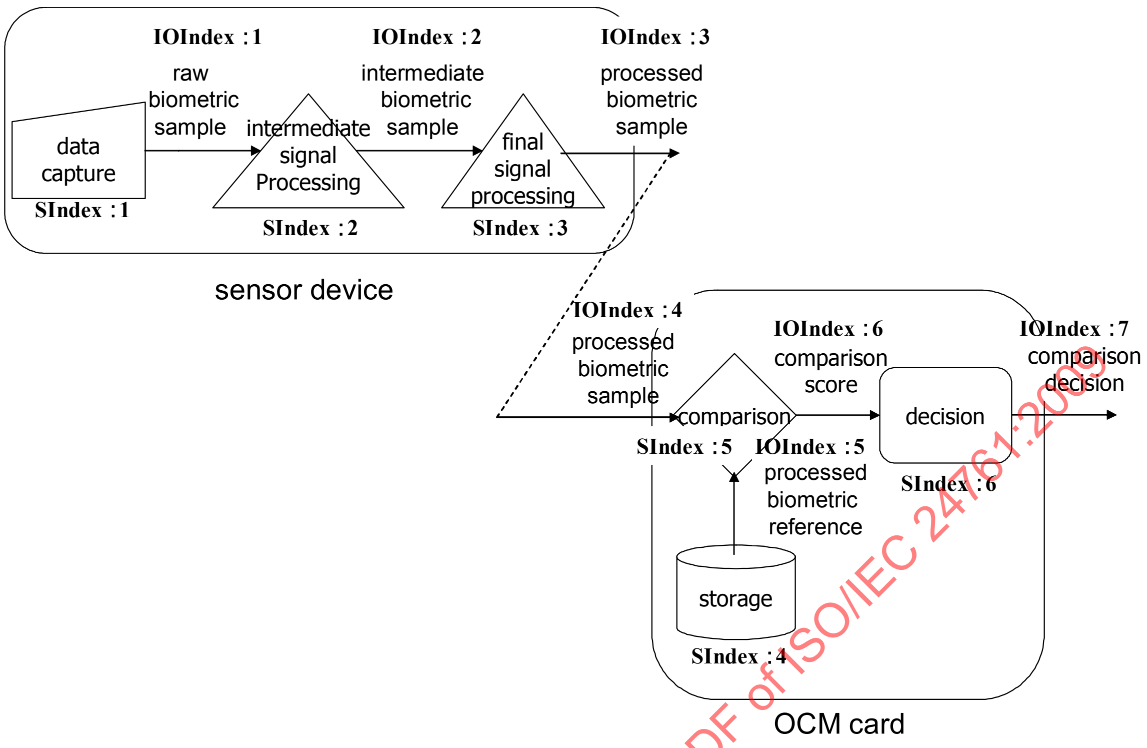
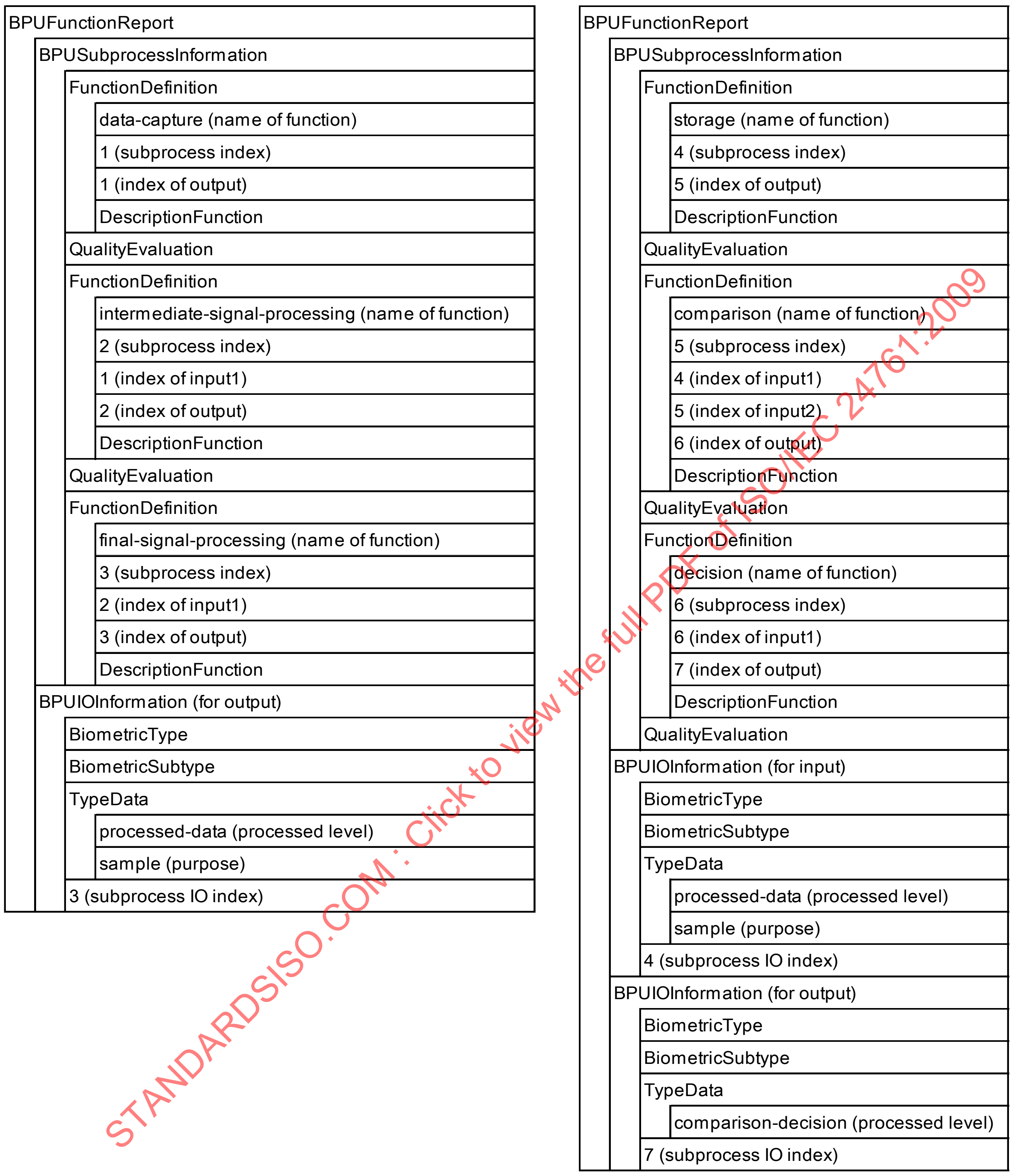


Figure B.8 — A biometric verification process of an OCM model and an example of indexing



BPUFunctionReport of the sensor device

BPUFunctionReport of the OCM card

Figure B.9 — examples of BPUFunctionReports for an OCM model