Federal Office
for Information Security

Technical Guideline BSI-TR-03129

PKIs for Machine Readable Travel Documents

Part 2: Supplemental specifications for public and official authorities

BSI-TR-03129-2
Version 1.3.0

Federal Office for Information Security
Post Box 20 03 63
D-53133 Bonn
Phone: +49 22899 9582-0
E-Mail: tr-03135@bsi.bund.de
Internet: https://www.bsi.bund.de
© Federal Office for Information Security 2017

Table of Contents

Tables

1 Introduction

This technical guideline, referenced as BSI-TR-03129-2, specifies PKI-related communication protocols for security mechanisms in the context of machine readable travel documents (MRTD). This document supplements BSI-TR-03129 with regard to document checks for public and official authorities.

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [1].

The key word “CONDITIONAL” is to be interpreted as follows: The usage of an item depends on the use of other items. It is therefore further qualified under which conditions the item is REQUIRED or RECOMMENDED.

Table 1: Key words

Parameter	**Meaning**
MUST / SHALL	REQUIRED
MUST NOT / SHALL NOT	-
SHOULD	RECOMMENDED
MAY	OPTIONAL
-	CONDITIONAL

2 Implemenation as Web Services

2.1 Web Service using SOAP messages

All protocols must be implemented as Web Services using SOAP messages as specified in BSI-TR-03129.

3 Protocols for Passive Authentication

Protocols and specifications for Terminal Authentication are used as specified in BSI-TR-03129. Beside SendMasterList and SendDefectList a TCC SHALL also implement `SendDocumentSignerList` and `GetDocumentSignerList` as described in the following sections.

3.1 Distribution of Document Signer Lists

The following messages are used for the distribution of Document Signer Lists.

3.1.1 GetDocumentSignerList

This message is sent by a terminal to its DV in order to get one or more signed lists of document signers.

Table 2: Input, Output and Return Codes parameters for message GetDocumentSignerList

Parameter	**Property**
Input	
`callbackIndicator`	REQUIRED
`messageID`	CONDITONAL
`responseURL`	CONDITONAL
Output	
`documentSignerList`	CONDITIONAL
Return Codes	The following return codes are supported - `ok_list_available` - `ok_syntax` - `ok_reception_ack` - `failure_list_not_available` - `failure_syntax` - `failure_synchronous_processing_not_possible` - `failure_internal_error`

3.1.2 SendDocumentSignerList

If the DV processes the message `GetDefectList` asynchronously, it uses the callback message `SendDefectList` to communicate the result of its processing. It sends the response message always to the URL specified in the parameter `responseURL` and the provided `messageID` of the received message.

This message can also be used to notify registered entities about the availability of a new Defect List. In this case the `messageID` MUST be omitted.

Table 3: Input, Output and Return Codes parameters for message SendDocumentSignerList

Parameter	Property
Input	
`messageID`	CONDITIONAL
`statusInfo`	REQUIRED The following status codes are supported - `ok_list_available` - `failure_list_not_available` - `failure_syntax` - `failure_internal_error`
`documentSignerList`	CONDITIONAL
Output	
Return Codes	The following return codes are supported - `ok_received_correctly` - `failure_syntax` - `failure_messageID_unknown` - `failure_internal_error`

3.2 Passive Authentication

Protocols for Passive Authentication are defined as specified in BSI-TR-03129. Please note that for compliance with BSI-TR-03129-2, in `GetDocumentSignerInformation` the parameter `DSInfo` contains an additional property, namely `dsCertificate`.

Table 4: DSInfo parameter definitions

Parameter	Property
`dsTrustStatus`	As definied in BSI-TR-03129.
`CertValidity`	As definied in BSI-TR-03129.
`dsSignature`	As definied in BSI-TR-03129.
`knownDefetcs`	As definied in BSI-TR-03129.
`cscaCertificate`	As definied in BSI-TR-03129.
`dsCertificate`	In case of `SignerIdentifier` being set in `GetDocumentSignerInformation`, this parameter contains the DS certificate which has been used to sign the Security Object, if available.
`unknownCriticalExtension`	As definied in BSI-TR-03129.

3.3 Protocols for Terminal Authentication

Protocols and specifications for Terminal Authentication are used as specified in BSI-TR-03129.

4 Defect Lists for ePassport Application

This chapter defines Defect Lists for the handling of Defects in the context of official document checks making use of the ePassport Application on a chip. Defects relating to the eID Application are defined in BSI-TR-03129 and not part of this document.

A defect is defined as a production error affecting large numbers of documents. The withdrawal of already issued documents is impractical or even impossible if the detected defect is contained in foreign documents.

Defect Lists are errata that not only inform about defects but also provides corrigenda to fix the error where possible. Defect Lists are signed lists to handle such defects. Defects are identified by the Document Signer Certificate(s) used to produce defect documents. Defect Lists are provided as Signed Data according to RFC3852 [4] and SHALL be using the profile and list content description as specified in BSI-TR-03129.

4.1 Defect List Format

A terminal compliant with BSI-TR-03129-2 SHALL support version 1 and version 2 of the Defect List format. A Defect List SHALL be provided as BER encoded `SignedData` with content type `DefectList` identified by `id-DefectList` (see BSI-TR-03129):

```
bsi-de OBJECT IDENTIFIER ::= {
  itu-t(0) identified-organization(4) etsi(0)
  reserved(127) etsi-identified-organization(0) 7
}

id-DefectList OBJECT IDENTIFIER ::= { bsi-de applications(3) mrtd(1) 5 }

DefectList ::= SEQUENCE {
                version INTEGER {v1(0), v2(1)},
                hashAlg OBJECT IDENTIFIER,
                defects SET OF Defect
}
```

Compared to version 1 of the Defect List Format, in version 2 of the Defect List Format a description field is added. It describes the class of a defect in human readable format and can contain additional information and notes on the defect or affected documents.

```
Defect ::= SEQUENCE {
        signerIdentifier SignerIdentifier OPTIONAL,    -- present in v1
        certificateHash OCTET STRING OPTIONAL,
        knownDefects SET OF VersionedKnownDefect,
        description UTF8String OPTIONAL                -- only present in v2
}
```

Like in `Defect` there is also a description field in KnownDefect. It describes the known defect with additional technical details.

```
VersionedKnownDefect ::= CHOICE {
        knownDefect       KnownDefect,         -- use this if version is v1
```

```
        knownDefectV2 [0] KnownDefectV2        -- use this if version is v2
}

KnownDefect ::= SEQUENCE {
        defectType OBJECT IDENTIFIER,
        parameters ANY DEFINED BY defectType OPTIONAL
}

KnownDefectV2 ::= SEQUENCE {
        defectType OBJECT IDENTIFIER,
        parameters  [0] ANY DEFINED BY defectType OPTIONAL,
        description [1] UTF8String OPTIONAL
}
```

The data type SignerIdentifier is defined in [4] as follows

```
SignerIdentifier ::= CHOICE {
        issuerAndSerialNumber IssuerAndSerialNumber,
        subjectKeyIdentifier [0] SubjectKeyIdentifier
}
```

4.1.1 Identification of Document Signer Certificates

There are two options to reference the Document Signer Certificate in a Defect List, either by the distinguished name of the CSCA and the serial number of the Document Signer Certificate or by the Subject Key Identifier of the Document Signer Public Key. Those two options are augmented by a third option: If neither issuerAndSerialNumber nor the subjectKeyIdentifier uniquely identify the Document Signer (due to a defect in the certificate) the hash of the Document Signer Certificate MUST additionally be included. The hash function to be used is indicated in the Defect List.

4.2 Defect Categories

Defects are categorized into three different types

- Authentication Defects
- Application Defects
- Document Defects

4.2.1 Authentication Defects

This section describes defects related to Passive Authentication, Chip Authentication, and Active Authentication. This category of defects is indicated by the following object identifier:

```
id-AuthDefect OBJECT IDENTIFIER ::= {id-DefectList 1}
```

4.2.1.1 Document Signer Certificate Revoked

The private key of the Document Signer is compromised (e.g. revoked by a CRL).

Note that this defect is equivalent to an issued CRL containing the Document Signer. This type of defect can be used to revoke foreign Document Signers independently from a CRL.

This type of defect is indicated by the following object identifier. It provides StatusCode as a parameter.

```
id-CertRevoked OBJECT IDENTIFIER ::= {id-AuthDefect 1}

StatusCode ::= ENUMERATED {
        noIndication(0),        -- no details given
        onHold(1),              -- revocation under investigation
        testing(2),             -- the certificate has been used for testing
        revokedByIssuer(3),     -- the Issuer has revoked the certificate by CRL
        revokedDLS(4),          -- DS has revoked the certificate
        certInadequate(5),      -- The certificate is inadequate
        proprietary(32)         -- status codes >=32 for internal use
}
```

4.2.1.2 Document Signer Certificate Malformed

The Document Signer Certificate is malformed and cannot be decoded correctly. Instead a replacement certificate is provided. Note that this certificate is not necessarily signed by the corresponding CSCA. Replacement Document Signer Certificates MAY be self-signed.

This type of defect is indicated by the following object identifier. It provides a replacement certificate (Certificate, cf. [5]) as parameter.

```
id-CertReplaced OBJECT IDENTIFIER ::= {id-AuthDefect 2}
```

4.2.1.3 Chip Authentication Private Keys Compromised

The Chip Authentication Private Keys have been compromised (e.g. due to a error in the key generation algorithm). This type of defect is indicated by the following object identifier. It does not provide any parameters:

```
id-ChipAuthKeyRevoked OBJECT IDENTIFIER ::= {id-AuthDefect 3}
```

4.2.1.4 Active Authentication Private Keys Compromised

The Active Authentication Private Keys have been compromised (e.g. due to a error in the key generation algorithm). This type of defect is indicated by the following object identifier. It does not provide any parameters:

```
id-ActiveAuthKeyRevoked OBJECT IDENTIFIER ::= {id-AuthDefect 4}
```

4.2.1.5 Authentication Protocol Failure

AA or CA cannot be performed correctly. This type of defect is indicated by the following object identifier, without StatusCode as return parameter:

```
id-AuthenticationProtocolFailure OBJECT IDENTIFIER ::= {id-AuthDefect 5}
StatusCode ::= ENUMERATED {
        CA_failure(0),          -- CA is known to fail
        AA_failure(1),          -- AA is known to fail
        CA_AA_failure(2)        -- CA and AA are known to fail
        PACE_CAM_failure(3)     -- PACE-CAM known to fail
}
```

4.2.1.6 Validity Period Incorrect

The validity period of the DS or CSCA is incorrect, i. e. the certificate-chain did not follow the PKIX shell model. This type of defect is indicated by the following object identifier. It provides StatusCode as a parameter.

```
id-ValidityPeriodIncorrect OBJECT IDENTIFIER ::= {id-AuthDefect 6}
StatusCode ::= ENUMERATED {
        CSCA_validity(0),       -- CSCA cert. validity is incorrect
        DS_validity(1)          -- DS cert. validity is incorrect
        CSCA_DS_validity(2),    -- Validity of both certs incorrect
}
```

4.2.2 Application Defects (for the ePassport Application)

The following sections describe defects related to the personalisation of the ePassport application. This category of defects is indicated by the following object identifier:

```
id-ePassportDefect OBJECT IDENTIFIER ::= {id-DefectList 2}
```

4.2.2.1 Data Group Malformed

The indicated data groups might be incorrectly encoded. This type of defect is indicated by the following object identifier. It provides the malformed datagroups with the parameter `MalformedDGs`:

```
id-ePassportDGMalformed OBJECT IDENTIFIER ::= {id-ePassportDefect 1}


MalformedDGs ::= SET OF INTEGER -- DGs as integer
```

4.2.2.2 Document Security Object Malformed

The validation of the Document Security Object might fail (e.g. signature incorrect). This type of defect is indicated by the following object identifier. It provides `SODFaultStatusCode` as an optional parameter.

```
id-SODInvalid OBJECT IDENTIFIER ::= {id-ePassportDefect 2}


SodFaultStatusCode ::= ENUMERATED {
        noIndication(0),                -- no details given
        sodSignatureFailure(1),         -- Failure in SOD's signature
```

```
        sodHashInvalid(2),              -- sodHashInvalid
        sodEncodingFailure(3),          -- wrong encoding / encoding error
        sodDGHashesInvalid(4),          -- wrong/invalid DG hashes (or encoding)
        proprietary(32)                 -- >=32 codes used for internal purposes
}
```

4.2.2.3 COM and SOD Discrepancy

The list of present datagroups contained in EF.COM and/or EF.SOD are not correct. The Defect also provides `dataGroupDiscrepancy` as an optional parameter. This type of defect is indicated by the following object identifier.

```
id-COMSODDiscrepancy OBJECT IDENTIFIER ::= {id-ePassportDefect 3}


DataGroupDiscrepancy ::= SEQUENCE {
        listedInCOM SET OF INTEGER,     -- data groups as defined through COM
        listedInSOD SET OF INTEGER      -- data groups as defined through SOD
}
```

The parameter dataGroupDiscrepancy defines which datagroups are present by its definition through EF.COM or EF.SOD, thus it is possible to trace and analyze the discrepancy defect in more detail. If the optional parameter is not present, the discrepancy is interpreted as a generic.

4.2.2.4 Wrong Signer Identifier

The signer identifier in EF.SOD is wrong, hence no document signer certificate can be found in the security object. The signer identifier structure is used to request the corresponding DS certificate. If there exists a known defect for this identifier the TCC responds with a document signer certificate (Certificate, cf. [5]) as parameter.

```
id-sodWrongSignerIdentifier OBJECT IDENTIFIER ::= {id-ePassportDefect 4}
```

4.2.2.5 Issuing Country Defect

The referenced issuing Country in the certificate was wrong, e.g. ICAO 3-letter code instead of ICAO 2-letter code [2], or a missing country, or the issuer used a wrong issuing country. The Defect provides `correctedIssuingCountry` as an returned parameter.

```
id-IssuingCountryDefect OBJECT IDENTIFIER ::= {id-ePassportDefect 5}


correctedIssuingCountry :: = PrintableString
```

4.2.3 General Document Defects

The following sections describe defects related to the document in general. This category of defects is indicated by the following object identifier:

```
id-DocumentDefect OBJECT IDENTIFIER ::= {id-DefectList 4}
```

If a document is affected by a general defect (i.e. the defect is contained the sub-tree of id-DocumentDefect) with unknown interpretation, the electronic part of the document SHOULD NOT be used.

4.2.3.1 Card Security Object Malformed

The Card Security Object is incorrectly encoded. A corrected Card Security Object is provided and SHOULD be used. This type of defect is indicated by the following object identifier. It provides a corrected Card Security Object (SecurityObject, cf. [3]) as parameter:

```
id-CardSecurityMalformed OBJECT IDENTIFIER ::= {id-DocumentDefect 1}
```

4.2.3.2 Chip Security Object Malformed

The Chip Security Object might be incorrectly encoded. The Card Security Object SHOULD be used instead. This type of defect is indicated by the following object identifier. It does not provide any parameter:

```
id-ChipSecurityMalformed OBJECT IDENTIFIER ::= {id-DocumentDefect 2}
```

4.2.3.3 Powerdown Required

The chip denies multiple successive authentication using the General Authentication Procedure. Either the reader MUST powerdown the chip or the document MUST be removed from the reader in between two authentications. This type of defect is indicated by the following object identifier. It does not provide any parameter:

```
id-PowerDownReq OBJECT IDENTIFIER ::= {id-DocumentDefect 3}
```

4.2.3.4 Document Signer Certificate incorrectly encoded or malformed

The Document Signer Certificate is incorrectly encoded or malformed and could not be used. Compared to the Defect `id-CertReplaced` there is no replacement certificate provided. Hence additional information about this Defect is provided with `DSMalformedInformation` as a parameter. This type of defect is indicated by the following object identifier and a parameter:

```
id-DSMalformed OBJECT IDENTIFIER ::= {id-DocumentDefect 4}


DsMalformedInformation ::= ENUMERATED {
        noIndication(0),        -- no details given
        unknownCryptoAlg(1),    -- unknown cryptographic algorithm
        encodingFailure(2),     -- wrong encoding / encoding error
        proprietary(32)         -- codes >=32 can be used for internal purposes
}
```

5 Signed Data Profile for Document Signer Lists

Document Signer Lists are provided as Signed Data according to RFC3852 [4] and SHALL be using the profile as defined in BSI-TR-03129, Appendix C. The value `eContentType` shall be `id-DocumentSignerList`.

5.1 Document Signer List Format

A Document Signer List must be provided as BER encoded SignedData with content type identified by `id-DocumentSignerList`. The list of Document Signer certificates is a set of x.509 certificates in BER encoding.

```
bsi-de OBJECT IDENTIFIER ::= {
  itu-t(0) identified-organization(4) etsi(0)
  reserved(127) etsi-identified-organization(0) 7
}

id-DocumentSignerList OBJECT IDENTIFIER ::= { bsi-de applications(3) mrtd(1) 6 }
```

6 WSDL and XML Scheme specifications

The herein additionally defined scheme specifications and WSDL descriptions are provided as separate files supplementing BSI-TR-03129.

7 Biblopgraphy

[1] RFC 2119: Key words for use in RFCs to indicate requirement levels, 1997

[2] ICAO Doc 9303 – Part 3, 7th Edition, 2015

[3] BSI-TR-03110: Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS token, 2015

[4] RFC 3852: Cryptographic Message Syntax (CMS), 2004

[5] RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, 2008