Network Working Group
Request for Comments: 5698
Category: Standards Track

T. Kunz
Fraunhofer SIT
S. Okunick
pawisda systems GmbH
U. Pordesch
Fraunhofer Gesellschaft

November 2009

Data Structure for the Security Suitability of Cryptographic Algorithms (DSSC)

Abstract

Since cryptographic algorithms can become weak over the years, it is necessary to evaluate their security suitability. When signing or verifying data, or when encrypting or decrypting data, these evaluations must be considered. This document specifies a data structure that enables an automated analysis of the security suitability of a given cryptographic algorithm at a given point of time, which may be in the past, the present, or the future.

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Table of Contents

1. Introduction 4
1.1. Motivation 4
1.2. Terminology 5
1.2.1. Conventions Used in This Document 5
1.3. Use Cases 5
2. Requirements and Assumptions 5
2.1. Requirements 6
2.2. Assumptions 6
3. Data Structures 7
3.1. SecuritySuitabilityPolicy 7
3.2. PolicyName 8
3.3. Publisher 9
3.4. PolicyIssueDate 9
3.5. NextUpdate 9
3.6. Usage 9
3.7. Algorithm 9
3.8. AlgorithmIdentifier 10
3.9. Evaluation 10
3.10. Parameter 11
3.11. Validity 12
3.12. Information 12
3.13. Signature 12
4. DSSC Policies 13
5. Definition of Parameters 13
6. Processing 14
6.1. Inputs 14
6.2. Verify Policy 14
6.3. Algorithm Evaluation 15
6.4. Evaluation of Parameters 15
6.5. Output 16
7. Security Considerations 16
8. IANA Considerations 18
9. References 23
9.1. Normative References 23
9.2. Informative References 24
Appendix A. DSSC and ERS 27
A.1. Verification of Evidence Records Using DSSC (Informative) 27
A.2. Storing DSSC Policies in Evidence Records (Normative) 27
Appendix B. XML Schema (Normative) 28
Appendix C. ASN.1 Module in 1988 Syntax (Informative) 30
Appendix D. ASN.1 Module in 1997 Syntax (Normative) 32
Appendix E. Example 34

1. Introduction

1.1. Motivation

Digital signatures can provide data integrity and authentication. They are based on cryptographic algorithms that are required to have certain security properties. For example, hash algorithms must be resistant to collisions, and in case of public key algorithms, computation of the private key that corresponds to a given public key must be infeasible. If algorithms lack the required properties, signatures could be forged, unless they are protected by a strong cryptographic algorithm.

Cryptographic algorithms that are used in signatures shall be selected to resist such attacks during their period of use. For signature keys included in public key certificates, this period of use is the validity period of the certificate. Cryptographic algorithms that are used for encryption shall resist such attacks during the period it is planned to keep the information confidential.

Only very few algorithms satisfy the security requirements. Besides, because of the increasing performance of computers and progresses in cryptography, algorithms or their parameters become insecure over the years. The hash algorithm MD5, for example, is unsuitable today for many purposes. A digital signature using a "weak" algorithm has no probative value, unless the "weak" algorithm has been protected by a strong algorithm before the time it was considered to be weak. Many kinds of digital signed data (including signed documents, timestamps, certificates, and revocation lists) are affected, particularly in the case of long-term archiving. Over long periods of time, it is assumed that the algorithms used in signatures become insecure.

For this reason, it is important to periodically evaluate an algorithm’s fitness and to consider the results of these evaluations when creating and verifying signatures, or when maintaining the validity of signatures made in the past. One result is a projected validity period for the algorithm, i.e., a prediction of the period of time during which the algorithm is fit for use. This prediction can help to detect whether a weak algorithm is used in a signature and whether that signature has been properly protected in due time by another signature made using an algorithm that is suitable at the present point of time. Algorithm evaluations are made by expert committees. In Germany, the Federal Network Agency annually publishes evaluations of cryptographic algorithms [BNetzAg.2008]. Examples of other European and international evaluations are [ETSI-TS102176-1-2005] and [NIST.800-57-Part1.2006].

These evaluations are published in documents intended to be read by humans. Therefore, to enable automated processing, it is necessary to define a data structure that expresses the content of the evaluations. This standardized data structure can be used for publication and can be interpreted by signature generation and verification tools. Algorithm evaluations are pooled in a security suitability policy. In this document, a data structure for a security suitability policy is specified. Therefore, the document provides a framework for expressing evaluations of cryptographic algorithms. This document does not attempt to catalog the security properties of cryptographic algorithms. Furthermore, no guidelines are made about which kind of algorithms shall be evaluated, for example, security suitability policies may be used to evaluate public key and hash algorithms, signature schemes, and encryption schemes.

1.2. Terminology

Algorithm: A cryptographic algorithm, i.e., a public key or hash algorithm. For public key algorithms, this is the algorithm with its parameters, if any. Furthermore, the term "algorithm" is used for cryptographic schemes and for actually padding functions.

Operator: Instance that uses and interprets a policy, e.g., a signature-verification component.

Policy: An abbreviation for security suitability policy.

Publisher: Instance that publishes the policy containing the evaluation of algorithms.

Security suitability policy: The evaluation of cryptographic algorithms with regard to their security in a specific application area, e.g., signing or verifying data. The evaluation is published in an electronic format.

Suitable algorithm: An algorithm that is evaluated against a policy and determined to be valid, i.e., resistant against attacks, at a particular point of time.

1.2.1. Conventions Used in This Document

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 [RFC2119].

1.3. Use Cases

Some use cases for a security suitability policy are presented here.

Long-term archiving: The most important use case is long-term archiving of signed data. Algorithms or their parameters become insecure over long periods of time. Therefore, signatures of archived data and timestamps have to be periodically renewed. A policy provides information about suitable and threatened algorithms. Additionally, the policy assists in verifying archived as well as re-signed documents.

Services: Services may provide information about cryptographic algorithms. On the basis of a policy, a service is able to provide the date when an algorithm became insecure or presumably will become insecure, as well as information regarding which algorithms are presently valid. Verification tools or long-term archiving systems can request such services and therefore do not need to deal with the algorithm security by themselves.

Long-term Archive Services (LTA) as defined in [RFC4810] may use the policy for signature renewal.

Signing and verifying: When signing documents or certificates, it must be assured that the algorithms used for signing or verifying are suitable. Accordingly, when verifying Cryptographic Message Syntax (CMS) [RFC5652] or XML signatures ([RFC3275], [ETSI-TS101903]), not only the validity of the certificates but also the validity of all involved algorithms may be checked.

Re-encryption: A security suitability policy can also be used to decide if encrypted documents must be re-encrypted because the encryption algorithm is no longer secure.

2. Requirements and Assumptions

Section 2.1 describes general requirements for a data structure containing the security suitability of algorithms. In Section 2.2, assumptions are specified concerning both the design and the usage of the data structure.

A policy contains a list of algorithms that have been evaluated by a publisher. An algorithm evaluation is described by its identifier, security constraints, and validity period. By these constraints, the requirements for algorithm properties must be defined, e.g., a public key algorithm is evaluated on the basis of its parameters.

2.1. Requirements

Automatic interpretation: The data structure of the policy must allow automated evaluation of the security suitability of an algorithm.

Flexibility: The data structure must be flexible enough to support new algorithms. Future policy publications may include evaluations of algorithms that are currently unknown. It must be possible to add new algorithms with the corresponding security constraints in the data structure. Additionally, the data structure must be independent of the intended use, e.g., encryption, signing, verifying, and signature renewing. Thus, the data structure is usable in every use case.

Source authentication: Policies may be published by different institutions, e.g., on the national or European Union (EU) level, whereas one policy needs not to be in agreement with the other one. Furthermore, organizations may undertake their own evaluations for internal purposes. For this reason a policy must be attributable to its publisher.

Integrity and authenticity: It must be possible to assure the integrity and authenticity of a published security suitability policy. Additionally, the date of issue must be identifiable.

2.2. Assumptions

It is assumed that a policy contains the evaluations of all currently known algorithms, including the expired ones.

An algorithm is suitable at a time of interest if it is contained in the current policy and the time of interest is within the validity period. Additionally, if the algorithm has any parameters, these parameters must meet the requirements defined in the security constraints.

If an algorithm appears in a policy for the first time, it may be assumed that the algorithm has already been suitable in the past. Generally, algorithms are used in practice prior to evaluation.

To avoid inconsistencies, multiple instances of the same algorithm are prohibited. The publisher must take care to prevent conflicts within a policy.

Assertions made in the policy are suitable at least until the next policy is published.

Publishers may extend the lifetime of an algorithm prior to reaching the end of the algorithm's validity period by publishing a revised policy. Publishers should not resurrect algorithms that are expired at the time a revised policy is published.

3. Data Structures

This section describes the syntax of a security suitability policy defined as an XML schema [W3C.REC-xmlschema-1-20041028]. ASN.1 modules are defined in Appendix C and Appendix D. The schema uses the following XML namespace [W3C.REC-xml-names-20060816]:

urn:ietf:params:xml:ns:dssc

Within this document, the prefix "dssc" is used for this namespace. The schema starts with the following schema definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:dssc="urn:ietf:params:xml:ns:dssc"
           xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
           targetNamespace="urn:ietf:params:xml:ns:dssc"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">
<xs:import namespace="http://www.w3.org/XML/1998/namespace"
           schemaLocation="http://www.w3.org/2001/xml.xsd"/>
<xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
           schemaLocation="xmldsig-core-schema.xsd"/>
```

3.1. SecuritySuitabilityPolicy

The SecuritySuitabilityPolicy element is the root element of a policy. It has an optional id attribute, which MUST be used as a reference when signing the policy (Section 3.13). The optional lang attribute defines the language according to [RFC5646]. The language is applied to all human-readable text within the policy. If the lang attribute is omitted, the default language is English ("en"). The element is defined by the following schema:

```
<xs:element name="SecuritySuitabilityPolicy"
            type="dssc:SecuritySuitabilityPolicyType"/>
<xs:complexType name="SecuritySuitabilityPolicyType">
  <xs:sequence>
    <xs:element ref="dssc:PolicyName"/>
    <xs:element ref="dssc:Publisher"/>
    <xs:element name="PolicyIssueDate" type="xs:dateTime"/>
    <xs:element name="NextUpdate" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="Usage" type="xs:string" minOccurs="0"/>
    <xs:element ref="dssc:Algorithm" maxOccurs="unbounded"/>
    <xs:element ref="ds:Signature" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="version" type="xs:string" default="1"/>
  <xs:attribute name="lang" default="en"/>
  <xs:attribute name="id" type="xs:ID"/>
</xs:complexType>
```

3.2. PolicyName

The PolicyName element contains an arbitrary name for the policy. The optional elements Object Identifier (OID) and Uniform Resource Identifier (URI) MAY be used for the identification of the policy. OIDs MUST be expressed in the dot notation.

```
<xs:element name="PolicyName" type="dssc:PolicyNameType"/>
<xs:complexType name="PolicyNameType">
  <xs:sequence>
    <xs:element ref="dssc:Name"/>
    <xs:element ref="dssc:ObjectIdentifier" minOccurs="0"/>
    <xs:element ref="dssc:URI" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:element name="Name" type="xs:string"/>
<xs:element name="ObjectIdentifier">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="(\d+\.)+\d+"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
<xs:element name="URI" type="xs:anyURI"/>
```

3.3. Publisher

The Publisher element contains information about the publisher of the policy. It is composed of the name (e.g., name of institution), an optional address, and an optional URI. The Address element contains arbitrary free-format text not intended for automatic processing.

```
<xs:element name="Publisher" type="dssc:PublisherType"/>
<xs:complexType name="PublisherType">
  <xs:sequence>
    <xs:element ref="dssc:Name"/>
    <xs:element name="Address" type="xs:string" minOccurs="0"/>
    <xs:element ref="dssc:URI" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

3.4. PolicyIssueDate

The PolicyIssueDate element indicates the point of time when the policy was issued.

3.5. NextUpdate

The optional NextUpdate element MAY be used to indicate when the next policy will be issued.

3.6. Usage

The optional Usage element determines the intended use of the policy (e.g., certificate validation, signing and verifying documents). The element contains free-format text intended only for human readability.

3.7. Algorithm

A security suitability policy MUST contain at least one Algorithm element. An algorithm is identified by an AlgorithmIdentifier element. Additionally, the Algorithm element contains all evaluations of the specific cryptographic algorithm. More than one evaluation may be necessary if the evaluation depends on the parameter constraints. The optional Information element MAY be used to provide additional information like references on algorithm specifications. In order to give the option to extend the Algorithm element, it additionally contains a wildcard. The Algorithm element is defined by the following schema:

```
<xs:element name="Algorithm" type="dssc:AlgorithmType"/>
<xs:complexType name="AlgorithmType">
  <xs:sequence>
    <xs:element ref="dssc:AlgorithmIdentifier"/>
    <xs:element ref="dssc:Evaluation" maxOccurs="unbounded"/>
    <xs:element ref="dssc:Information" minOccurs="0"/>
    <xs:any namespace="##other" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

3.8. AlgorithmIdentifier

The AlgorithmIdentifier element is used to identify a cryptographic algorithm. It consists of the algorithm name, at least one OID, and optional URIs. The algorithm name is not intended to be parsed by automatic processes. It is only intended to be read by humans. The OID MUST be expressed in dot notation (e.g., "1.3.14.3.2.26"). The element is defined as follows:

```
<xs:element name="AlgorithmIdentifier"
            type="dssc:AlgorithmIdentifierType"/>
<xs:complexType name="AlgorithmIdentifierType">
  <xs:sequence>
    <xs:element ref="dssc:Name"/>
    <xs:element ref="dssc:ObjectIdentifier" maxOccurs="unbounded"/>
    <xs:element ref="dssc:URI" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

3.9. Evaluation

The Evaluation element contains the evaluation of one cryptographic algorithm in dependence of its parameter constraints. For example, the suitability of the RSA algorithm depends on the modulus length (RSA with a modulus length of 1024 may have another suitability period as RSA with a modulus length of 2048). Current hash algorithms like SHA-1 or RIPEMD-160 do not have any parameters. Therefore, the Parameter element is optional. The suitability of the algorithm is expressed by a validity period, which is defined by the Validity element. An optional wildcard MAY be used to extend the Evaluation element.

```
<xs:element name="Evaluation" type="dssc:EvaluationType"/>
<xs:complexType name="EvaluationType">
  <xs:sequence>
    <xs:element ref="dssc:Parameter" minOccurs="0"
                                     maxOccurs="unbounded"/>
    <xs:element ref="dssc:Validity"/>
    <xs:any namespace="##other" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

3.10. Parameter

The Parameter element is used to express constraints on algorithm-specific parameters.

The Parameter element has a name attribute, which holds the name of the parameter (e.g., "moduluslength" for RSA [RFC3447]). Section 5 defines parameter names for currently known public key algorithms; these parameter names SHOULD be used. For the actual parameter, a range of values or an exact value may be defined. These constraints are expressed by the following elements:

Min: The Min element defines the minimum value of the parameter. That means values equal or greater than the given value meet the requirements.

Max: The Max element defines the maximum value the parameter may take.

At least one of both elements MUST be set to define a range of values. A range MAY also be specified by a combination of both elements, whereas the value of the Min element MUST be less than or equal to the value of the Max element. The parameter may have any value within the defined range, including the minimum and maximum values. An exact value is expressed by using the same value in both the Min and the Max element.

These constraints are sufficient for all current algorithms. If future algorithms need constraints that cannot be expressed by the elements above, an arbitrary XML structure MAY be inserted that meets the new constraints. For this reason, the Parameter element contains a wildcard. A parameter MUST contain at least one constraint. The schema for the Parameter element is as follows:

```
<xs:element name="Parameter" type="dssc:ParameterType"/>
<xs:complexType name="ParameterType">
  <xs:sequence>
    <xs:element name="Min" type="xs:int" minOccurs="0"/>
    <xs:element name="Max" type="xs:int" minOccurs="0"/>
    <xs:any namespace="##other" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required"/>
</xs:complexType>
```

3.11. Validity

The Validity element is used to define the period of the (predicted) suitability of the algorithm. It is composed of an optional start date and an optional end date. Defining no end date means the algorithm has an open-end validity. Of course, this may be restricted by a future policy that sets an end date for the algorithm. If the end of the validity period is in the past, the algorithm was suitable until that end date. The element is defined by the following schema:

```
<xs:element name="Validity" type="dssc:ValidityType"/>
<xs:complexType name="ValidityType">
  <xs:sequence>
    <xs:element name="Start" type="xs:date" minOccurs="0"/>
    <xs:element name="End" type="xs:date" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

3.12. Information

The Information element MAY be used to give additional textual information about the algorithm or the evaluation, e.g., references on algorithm specifications. The element is defined as follows:

```
<xs:element name="Information" type="dssc:InformationType"/>
<xs:complexType name="InformationType">
  <xs:sequence>
    <xs:element name="Text" type="xs:string" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

3.13. Signature

The optional Signature element MAY be used to guarantee the integrity and authenticity of the policy. It is an XML signature specified in [RFC3275]. The signature MUST relate to the SecuritySuitabilityPolicy element. If the Signature element is set, the SecuritySuitabilityPolicy element MUST have the optional id attribute. This attribute MUST be used to reference the SecuritySuitabilityPolicy element within the Signature element. Since it is an enveloped signature, the signature MUST use the transformation algorithm identified by the following URI:

http://www.w3.org/2000/09/xmldsig#enveloped-signature

4. DSSC Policies

DSSC policies MUST be expressed either in XML or ASN.1. However, in order to reach interoperability, DSSC policies SHOULD be published in both XML and ASN.1.

In the case of XML, a DSSC policy is an XML document that MUST be well-formed and SHOULD be valid. XML-encoded DSSC policies MUST be based on XML 1.0 [W3C.REC-xml-20081126] and MUST be encoded using UTF-8 [RFC3629]. This specification makes use of XML namespaces [W3C.REC-xml-names-20060816] for identifying DSSC policies. The namespace URI for elements defined by this specification is a URN [RFC2141] using the namespace prefix "dssc". This URN is:

urn:ietf:params:xml:ns:dssc

XML-encoded DSSC policies are identified with the MIME type "application/dssc+xml" and are instances of the XML schema [W3C.REC-xmlschema-1-20041028] defined in Appendix B.

A file containing a DSSC policy in ASN.1 representation (for specification of ASN.1 refer to [CCITT.x208.1988], [CCITT.x209.1988], [CCITT.x680.2002] and [CCITT.x690.2002]) MUST contain only the DER encoding of one DSSC policy, i.e., there MUST NOT be extraneous header or trailer information in the file. ASN.1-based DSSC policies are identified with the MIME type "application/dssc+der". Appropriate ASN.1 modules are defined in Appendices C (1988-ASN.1 syntax) and D (1997-ASN.1 syntax).

5. Definition of Parameters

This section defines the parameter names for the currently known public key algorithms. The following parameters also refer to cryptographic schemes based on these public key algorithms (e.g., the PKCS#1 v1.5 signature scheme SHA-256 with RSA [RFC3447]).

The parameter of RSA [RFC3447] SHOULD be named "moduluslength".

The parameters for the Digital Signature Algorithm (DSA) [FIPS186-2] SHOULD be "plength" and "qlength".

These parameter names have been registered by IANA (see Section 8). It may be necessary to register further algorithms not given in this section (in particular, future algorithms). The process for registering parameter names of further algorithms is described in Section 8. Publishers of policies SHOULD use these parameter names so that the correct interpretation is guaranteed.

6. Processing

Evaluation of an algorithm’s security suitability is described in three parts: verification of the policy, determination of algorithm validity, and evaluation of algorithm parameters, if any.

In the following sections, a process is described

- to determine if an algorithm was suitable at a particular point of time, and
- to determine until what time an algorithm was or will be suitable.

6.1. Inputs

To determine the security suitability of an algorithm, the following information is required:

- Policy
- Current time
- Algorithm identifier and parameter constraints (if associated)
- Time of interest (optional). Providing no time of interest means determination of the validity end date of the algorithm.

6.2. Verify Policy

The signature on the policy SHOULD be verified and a certification path from the policy signer's certificate to a current trust anchor SHOULD be constructed and validated [RFC5280]. The algorithms used to verify the digital signature and validate the certification path MUST be suitable per the contents of the policy being verified. If signature verification fails, certification path validation fails or an unsuitable algorithm is required to perform these checks, then the policy MUST be rejected.

The nextUpdate time in the policy MUST be either greater than the current time or absent. If the nextUpdate time is less than the current time, the policy MUST be rejected.

6.3. Algorithm Evaluation

To determine the validity period of an algorithm, locate the Algorithm element in the policy that corresponds to the algorithm identifier provided as input. The Algorithm element is located by comparing the OID in the element to the OID included in the algorithm identifier provided as input.

If no matching Algorithm element is found, then the algorithm is unknown.

If the time of interest was provided as input, the validity of each Evaluation element MUST be checked in order to determine if the algorithm was suitable at the time of interest. For each Evaluation element:

- Confirm the Start time is either less than the time of interest or absent. Discard the entry if the Start time is present and greater than the time of interest.

- Confirm the End time is either greater than the time of interest or absent. Discard the entry if the End time is present and less than the time of interest.

If all Evaluation elements were rejected, the algorithm is not suitable according to the policy.

Any entries not rejected will be used for the evaluation of the parameters, if any.

6.4. Evaluation of Parameters

Any necessary parameters of the entries not rejected MUST be evaluated within the context of the type and usage of the algorithm. Details of parameter evaluation are defined on a per-algorithm basis.

To evaluate the parameters, the Parameter elements of each Evaluation element that has not been rejected in the process described in Section 6.3 MUST be checked. For each Parameter element:

- Confirm that the parameter was provided as input. Discard the Evaluation element if the parameter does not match to any of the parameters provided as input.
- If the Parameter element has a Min element, confirm that the parameter value is less than or equal to the corresponding parameter provided as input. Discard the Evaluation element if the parameter value does not meet the constraint.
- If the Parameter element has a Max element, confirm that the parameter value is greater than or equal to the corresponding parameter provided as input. Discard the Evaluation element if the parameter value does not meet the constraint.
- If the Parameter has another constraint, confirm that the value of the corresponding parameter provided as input meets this constraint. If it does not or if the constraint is unrecognized, discard the Evaluation element.

If all Evaluation elements were rejected, the algorithm is not suitable according to the policy.

Any entries not rejected will be provided as output.

6.5. Output

If the algorithm is not in the policy, return an error "algorithm unknown".

If no time of interest was provided as input, return the maximum End time of the Evaluation elements that were not discarded. If at least one End time of these Evaluation elements is absent, return "algorithm has an indefinite End time".

Otherwise, if the algorithm is not suitable relative to the time of interest, return an error "algorithm unsuitable".

If the algorithm is suitable relative to the time of interest, return the Evaluation elements that were not discarded.

7. Security Considerations

The policy for an algorithm's security suitability has a great impact on the quality of the results of signature generation and verification operations. If an algorithm is incorrectly evaluated against a policy, signatures with a low probative force could be created or verification results could be incorrect. The following security considerations have been identified:

1. Publishers MUST ensure unauthorized manipulation of any security suitability is not possible prior to a policy being signed and published. There is no mechanism provided to revoke a policy after publication. Since the algorithm evaluations change infrequently, the lifespan of a policy should be carefully considered prior to publication.

2. Operators SHOULD only accept policies issued by a trusted publisher. Furthermore, the validity of the certificate used to sign the policy SHOULD be verifiable by Certificate Revocation List (CRL) [RFC5280] or Online Certificate Status Protocol (OCSP) [RFC2560]. The certificate used to sign the policy SHOULD be revoked if the algorithms used in this certificate are no longer suitable. It MUST NOT be possible to alter or replace a policy once accepted by an operator.

3. Operators SHOULD periodically check to see if a new policy has been published to avoid using obsolete policy information. For publishers, it is suggested not to omit the NextUpdate element in order to give operators a hint regarding when the next policy will be published.

4. When signing a policy, algorithms that are suitable according to this policy SHOULD be used.

5. The processing rule described in Section 6 is about one cryptographic algorithm independent of the use case. Depending upon the use case, an algorithm that is no longer suitable at the time of interest, does not necessarily mean that the data structure where it is used is no longer secure. For example, a signature has been made with an RSA signer's key of 1024 bits. This signature is timestamped with a timestamp token that uses an RSA key of 2048 bits, before an RSA key size of 1024 bits will be broken. The fact that the signature key of 1024 bits is no longer suitable at the time of interest does not mean that the

whole data structure is no longer secure, if an RSA key size of 2048 bits is still suitable at the time of interest.

6. In addition to the key size considerations, other considerations must be applied, like whether a timestamp token has been provided by a trusted authority. This means that the simple use of a suitability policy is not the single element to consider when evaluating the security of a complex data structure that uses several cryptographic algorithms.

7. The policies described in this document are suitable to evaluate basic cryptographic algorithms, like public key or hash algorithms, as well as cryptographic schemes (e.g., the PKCS#1 v1.5 signature schemes [RFC3447]). But it MUST be kept in mind that a basic cryptographic algorithm that is suitable according to the policy does not necessarily mean that any cryptographic schemes based on this algorithm are also secure. For example, a signature scheme based on RSA must not necessarily be secure if RSA is suitable. In case of a complete signature verification, including validation of the certificate path, various algorithms have to be checked against the policy (i.e., signature schemes of signed data objects and revocation information, public key algorithms of the involved certificates, etc.). Thus, a policy SHOULD contain evaluations of public key and hash algorithms as well as of signature schemes.

8. Re-encrypting documents that were originally encrypted using an algorithm that is no longer suitable will not protect the semantics of the document if the document has been intercepted. However, for documents stored in an encrypted form, re-encryption must be considered, unless the document has lost its original value.

8. IANA Considerations

This document defines the XML namespace "urn:ietf:params:xml:ns:dssc" according to the guidelines in [RFC3688]. This namespace has been registered in the IANA XML Registry.

This document defines an XML schema (see Appendix B) according to the guidelines in [RFC3688]. This XML schema has been registered in the IANA XML Registry and can be identified with the URN "urn:ietf:params:xml:schema:dssc".

This document defines the MIME type "application/dssc+xml". This MIME type has been registered by IANA under "MIME Media Types" according to the procedures of [RFC4288].

Type name: application

Subtype name: dssc+xml

Required parameters: none

Optional parameters: "charset" as specified for "application/xml" in [RFC3023].

Encoding considerations: Same as specified for "application/xml" in [RFC3023].

Security considerations: Same as specified for "application/xml" in Section 10 of [RFC3023]. For further security considerations, see Section 7 of this document.

Interoperability considerations: Same as specified for "application/xml" in [RFC3023].

Published specification: This document.

Applications that use this media: Applications for long-term archiving of signed data, applications for signing data / verifying signed data, and applications for encrypting / decrypting data.

Additional information:

Magic number(s): none

File extension(s): .xdssc

Macintosh file type code: "TEXT"

Object Identifiers: none

Person to contact for further information: Thomas Kunz (thomas.kunz@sit.fraunhofer.de)

Intended usage: COMMON

Restrictions on usage: none

Author/Change controller: IETF

This document defines the MIME type "application/dssc+der". This MIME type has been registered by IANA under "MIME Media Types" according to the procedures of [RFC4288].

Type name: application

Subtype name: dssc+der

Required parameters: none

Optional parameters: none

Encoding considerations: binary

Security considerations: See Section 7 of this document.

Interoperability considerations: none

Published specification: This document.

Applications that use this media: Applications for long-term archiving of signed data, applications for signing data / verifying signed data, and applications for encrypting / decrypting data.

Additional information:

Magic number(s): none

File extension(s): .dssc

Macintosh file type code: none

Object Identifiers: none

Person to contact for further information: Thomas Kunz (thomas.kunz@sit.fraunhofer.de)

Intended usage: COMMON

Restrictions on usage: none

Author/Change controller: IETF

This specification creates a new IANA registry entitled "Data Structure for the Security Suitability of Cryptographic Algorithms (DSSC)". This registry contains two sub-registries entitled "Parameter Definitions" and "Cryptographic Algorithms". The policy for future assignments to the sub-registry "Parameter Definitions" is "RFC Required".

The initial values for the "Parameter Definitions" sub-registry are:

Value	Description	Reference
moduluslength	Parameter for RSA (integer value)	RFC 5698
plength	Parameter for DSA (integer value, used together with parameter "qlength")	RFC 5698
qlength	Parameter for DSA (integer value, used together with parameter "plength")	RFC 5698

The sub-registry "Cryptographic Algorithms" contains textual names as well as Object Identifiers (OIDs) and Uniform Resource Identifiers (URIs) of cryptographic algorithms. It serves as assistance when creating a new policy. The policy for future assignments is "First Come First Served". When registering a new algorithm, the following information MUST be provided:

- The textual name of the algorithm.
- The OID of the algorithm.
- A reference to a publicly available specification that defines the algorithm and its identifiers.

Optionally, a URI MAY be provided if possible.

The initial values for the "Cryptographic Algorithms" sub-registry are:

Name	OID / URI	Reference
rsaEncryption	1.2.840.113549.1.1.1	[RFC3447]
dsa	1.2.840.10040.4.1	[RFC3279]
md2	1.2.840.113549.2.2	[RFC3279]
md5	1.2.840.113549.2.5	[RFC3279]
	http://www.w3.org/2001/04/xmldsig-more#md5	[RFC4051]
sha-1	1.3.14.3.2.26	[RFC3279]
	http://www.w3.org/2000/09/xmldsig#sha1	[RFC3275]
sha-224	2.16.840.1.101.3.4.2.4	[RFC4055]
	http://www.w3.org/2001/04/xmldsig-more#sha224	[RFC4051]
sha-256	2.16.840.1.101.3.4.2.1	[RFC4055]
sha-384	2.16.840.1.101.3.4.2.2	[RFC4055]
	http://www.w3.org/2001/04/xmldsig-more#sha384	[RFC4051]
sha-512	2.16.840.1.101.3.4.2.3	[RFC4055]
md2WithRSAEncryption	1.2.840.113549.1.1.2	[RFC3443]
md5WithRSAEncryption	1.2.840.113549.1.1.4	[RFC3443]
	http://www.w3.org/2001/04/xmldsig-more#rsa-md5	[RFC4051]
sha1WithRSAEncryption	1.2.840.113549.1.1.5	[RFC3443]
	http://www.w3.org/2000/09/xmldsig#rsa-sha1	[RFC3275]
sha256WithRSAEncryption	1.2.840.113549.1.1.11	[RFC3443]
	http://www.w3.org/2001/04/xmldsig-more#rsa-sha256	[RFC4051]
sha384WithRSAEncryption	1.2.840.113549.1.1.12	[RFC3443]
	http://www.w3.org/2001/04/xmldsig-more#rsa-sha384	[RFC4051]
sha512WithRSAEncryption	1.2.840.113549.1.1.13	[RFC3443]
	http://www.w3.org/2001/04/xmldsig-more#rsa-sha512	[RFC4051]
sha1WithDSA	1.2.840.10040.4.3	[RFC3279]
	http://www.w3.org/2000/09/xmldsig#dsa-sha1	[RFC3275]

9. References

9.1. Normative References

[CCITT.x680.2002]
International Telephone and Telegraph Consultative Committee, "Abstract Syntax Notation One (ASN.1): Specification of basic notation", CCITT Recommendation X.680, July 2002.

[CCITT.x690.2002]
International Telephone and Telegraph Consultative Committee, "AASN.1 encoding rules: Specification of basic encoding Rules (BER), Canonical encoding rules (CER) and Distinguished encoding rules (DER)", CCITT Recommendation X.690, July 2002.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2141] Moats, R., "URN Syntax", RFC 2141, May 1997.

[RFC2560] Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 2560, June 1999.

[RFC3023] Murata, M., St. Laurent, S., and D. Kohn, "XML Media Types", RFC 3023, January 2001.

[RFC3275] Eastlake, D., Reagle, J., and D. Solo, "(Extensible Markup Language) XML-Signature Syntax and Processing", RFC 3275, March 2002.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC4288] Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", BCP 13, RFC 4288, December 2005.

[RFC4998] Gondrom, T., Brandner, R., and U. Pordesch, "Evidence Record Syntax (ERS)", RFC 4998, August 2007.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5646] Phillips, A. and M. Davis, "Tags for Identifying Languages", BCP 47, RFC 5646, September 2009.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 5652, September 2009.

[W3C.REC-xml-20081126] Yergeau, F., Maler, E., Paoli, J., Sperberg-McQueen, C., and T. Bray, "Extensible Markup Language (XML) 1.0 (Fifth Edition)", World Wide Web Consortium Recommendation REC-xml-20081126, November 2008, <http://www.w3.org/TR/2008/REC-xml-20081126>.

[W3C.REC-xml-names-20060816] Layman, A., Hollander, D., Tobin, R., and T. Bray, "Namespaces in XML 1.0 (Second Edition)", World Wide Web Consortium Recommendation REC-xml-names-20060816, August 2006, <http://www.w3.org/TR/2006/REC-xml-names-20060816>.

[W3C.REC-xmlschema-1-20041028] Thompson, H., Beech, D., Mendelsohn, N., and M. Maloney, "XML Schema Part 1: Structures Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-1-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028>.

9.2. Informative References

[BNetzAg.2008] Federal Network Agency for Electricity, Gas, Telecommunications, Post and Railway, "Bekanntmachung zur elektronischen Signatur nach dem Signaturgesetz und der Signaturverordnung (Uebersicht ueber geeignete Algorithmen)", December 2007, <http://www.bundesnetzagentur.de/media/archive/12198.pdf>.

[CCITT.x208.1988] International Telephone and Telegraph Consultative Committee, "Specification of Abstract Syntax Notation One (ASN.1)", CCITT Recommendation X.208, November 1988.

[CCITT.x209.1988]
International Telephone and Telegraph Consultative Committee, "Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)", CCITT Recommendation X.209, November 1988.

[ETSI-TS101903]
European Telecommunication Standards Institute (ETSI), "XML Advanced Electronic Signatures (XAdES)", ETSI TS 101 903 V1.3.2, March 2006.

[ETSI-TS102176-1-2005]
European Telecommunication Standards Institute (ETSI), "Electronic Signatures and Infrastructures (ESI); "Algorithms and Parameters for Secure Electronic Signatures; Part 1: Hash functions and asymmetric algorithms"", ETSI TS 102 176-1 V2.0.0, November 2007.

[FIPS186-2]
National Institute of Standards and Technology, "Digital Signature Standard (DSS)", FIPS PUB 186-2 with Change Notice, January 2000.

[NIST.800-57-Part1.2006]
National Institute of Standards and Technology, "Recommendation for Key Management - Part 1: General (Revised)", NIST 800-57 Part 1, May 2006.

[RFC3279] Bassham, L., Polk, W., and R. Housley, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3279, April 2002.

[RFC3443] Agarwal, P. and B. Akyol, "Time To Live (TTL) Processing in Multi-Protocol Label Switching (MPLS) Networks", RFC 3443, January 2003.

[RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, February 2003.

[RFC4051] Eastlake, D., "Additional XML Security Uniform Resource Identifiers (URIs)", RFC 4051, April 2005.

[RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, June 2005.

[RFC4810] Wallace, C., Pordesch, U., and R. Brandner, "Long-Term Archive Service Requirements", RFC 4810, March 2007.

Appendix A. DSSC and ERS

A.1. Verification of Evidence Records Using DSSC (Informative)

This section describes the verification of an Evidence Record according to the Evidence Record Syntax (ERS, [RFC4998]), using the presented data structure.

An Evidence Record contains a sequence of ArchiveTimeStampChains, which consist of ArchiveTimeStamps. For each ArchiveTimeStamp the hash algorithm used for the hash tree (digestAlgorithm) as well as the public key algorithm and hash algorithm in the timestamp signature have to be examined. The relevant date is the time information in the timestamp (date of issue). Starting with the first ArchiveTimeStamp, it has to be assured that:

1. The timestamp uses public key and hash algorithms that were suitable at the date of issue.

2. The hashtree was built with a hash algorithm that was suitable at the date of issue as well.

3. Algorithms for timestamp and hashtree in the preceding ArchiveTimeStamp must have been suitable at the issuing date of considered ArchiveTimeStamp.

4. Algorithms in the last ArchiveTimeStamp have to be suitable now.

If the check of one of these items fails, this will lead to a failure of the verification.

A.2. Storing DSSC Policies in Evidence Records (Normative)

This section describes how to store a policy in an Evidence Record. ERS provides the field cryptoInfos for the storage of additional verification data. For the integration of a security suitability policy in an Evidence Record, the following content types are defined for both ASN.1 and XML representation:

```
DSSC_ASN1 {iso(1) identified-organization(3) dod(6)
        internet(1) security(5) mechanisms(5)
        ltans(11) id-ct(1) id-ct-dssc-asn1(2) }

DSSC_XML {iso(1) identified-organization(3) dod(6)
        internet(1) security(5) mechanisms(5)
        ltans(11) id-ct(1) id-ct-dssc-xml(3) }
```

Appendix B. XML Schema (Normative)

```
  <?xml version="1.0" encoding="UTF-8"?>
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
             xmlns:dssc="urn:ietf:params:xml:ns:dssc"
             xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
             targetNamespace="urn:ietf:params:xml:ns:dssc"
             elementFormDefault="qualified"
             attributeFormDefault="unqualified">
    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
               schemaLocation="http://www.w3.org/2001/xml.xsd"/>
    <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
               schemaLocation="xmldsig-core-schema.xsd"/>
    <xs:element name="SecuritySuitabilityPolicy"
                type="dssc:SecuritySuitabilityPolicyType"/>
    <xs:complexType name="SecuritySuitabilityPolicyType">
      <xs:sequence>
        <xs:element ref="dssc:PolicyName"/>
        <xs:element ref="dssc:Publisher"/>
        <xs:element name="PolicyIssueDate" type="xs:dateTime"/>
        <xs:element name="NextUpdate" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="Usage" type="xs:string" minOccurs="0"/>
        <xs:element ref="dssc:Algorithm" maxOccurs="unbounded"/>
        <xs:element ref="ds:Signature" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="version" type="xs:string" default="1"/>
      <xs:attribute name="lang" default="en"/>
      <xs:attribute name="id" type="xs:ID"/>
    </xs:complexType>
    <xs:element name="PolicyName" type="dssc:PolicyNameType"/>
    <xs:complexType name="PolicyNameType">
      <xs:sequence>
        <xs:element ref="dssc:Name"/>
        <xs:element ref="dssc:ObjectIdentifier" minOccurs="0"/>
        <xs:element ref="dssc:URI" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name="Publisher" type="dssc:PublisherType"/>
    <xs:complexType name="PublisherType">
      <xs:sequence>
        <xs:element ref="dssc:Name"/>
        <xs:element name="Address" type="xs:string" minOccurs="0"/>
        <xs:element ref="dssc:URI" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name="Name" type="xs:string"/>
    <xs:element name="ObjectIdentifier">
      <xs:simpleType>
```

```
        <xs:restriction base="xs:string">
          <xs:pattern value="(\d+\.)+\d+"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
    <xs:element name="URI" type="xs:anyURI"/>
    <xs:element name="Algorithm" type="dssc:AlgorithmType"/>
    <xs:complexType name="AlgorithmType">
      <xs:sequence>
        <xs:element ref="dssc:AlgorithmIdentifier"/>
        <xs:element ref="dssc:Evaluation" maxOccurs="unbounded"/>
        <xs:element ref="dssc:Information" minOccurs="0"/>
        <xs:any namespace="##other" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name="AlgorithmIdentifier"
                type="dssc:AlgorithmIdentifierType"/>
    <xs:complexType name="AlgorithmIdentifierType">
      <xs:sequence>
        <xs:element ref="dssc:Name"/>
        <xs:element ref="dssc:ObjectIdentifier" maxOccurs="unbounded"/>
        <xs:element ref="dssc:URI" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name="Validity" type="dssc:ValidityType"/>
    <xs:complexType name="ValidityType">
      <xs:sequence>
        <xs:element name="Start" type="xs:date" minOccurs="0"/>
        <xs:element name="End" type="xs:date" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name="Information" type="dssc:InformationType"/>
    <xs:complexType name="InformationType">
      <xs:sequence>
        <xs:element name="Text" type="xs:string" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name="Evaluation" type="dssc:EvaluationType"/>
    <xs:complexType name="EvaluationType">
      <xs:sequence>
        <xs:element ref="dssc:Parameter" minOccurs="0"
                                         maxOccurs="unbounded"/>
        <xs:element ref="dssc:Validity"/>
        <xs:any namespace="##other" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name="Parameter" type="dssc:ParameterType"/>
    <xs:complexType name="ParameterType">
```

```
      <xs:sequence>
        <xs:element name="Min" type="xs:int" minOccurs="0"/>
        <xs:element name="Max" type="xs:int" minOccurs="0"/>
        <xs:any namespace="##other" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="name" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:schema>
```

Appendix C. ASN.1 Module in 1988 Syntax (Informative)

```
   ASN.1-Module

   DSSC {iso(1) identified-organization(3) dod(6)
            internet(1) security(5) mechanisms(5)
            ltans(11) id-mod(0) id-mod-dssc88(6) id-mod-dssc88-v1(1) }

   DEFINITIONS IMPLICIT TAGS ::=
   BEGIN

   -- EXPORT ALL --

   IMPORTS

   -- Import from RFC 5280 [RFC5280]
   -- Delete following import statement
   -- if "new" types are supported

   UTF8String FROM PKIX1Explicit88
               { iso(1) identified-organization(3) dod(6)
               internet(1) security(5) mechanisms(5) pkix(7)
               mod(0) pkix1-explicit(18) }

   -- Import from RFC 5652 [RFC5652]

   ContentInfo FROM CryptographicMessageSyntax2004
               { iso(1) member-body(2) us(840)
               rsadsi(113549) pkcs(1) pkcs-9(9)
               smime(16) modules(0) cms-2004(24)}

   ;

   SecuritySuitabilityPolicy ::= ContentInfo

   -- contentType is id-signedData as defined in [RFC5652]
   -- content is SignedData as defined in [RFC5652]
   -- eContentType within SignedData is id-ct-dssc
   -- eContent within SignedData is TBSPolicy
```

```
   id-ct-dssc  OBJECT IDENTIFIER ::= {
               iso(1) identified-organization(3) dod(6)
               internet(1) security(5) mechanisms(5)
               ltans(11) id-ct(1) id-ct-dssc-tbsPolicy(6) }

   TBSPolicy ::= SEQUENCE {
        version          INTEGER               DEFAULT   {v1(1)},
        language         UTF8String            DEFAULT   "en",
        policyName       PolicyName,
        publisher        Publisher,
        policyIssueDate  GeneralizedTime,
        nextUpdate       GeneralizedTime       OPTIONAL,
        usage            UTF8String            OPTIONAL,
        algorithms       SEQUENCE OF Algorithm
   }

   PolicyName ::= SEQUENCE {
        name  UTF8String,
        oid   OBJECT IDENTIFIER OPTIONAL,
        uri   IA5String         OPTIONAL
   }

   Publisher ::= SEQUENCE {
        name        UTF8String,
        address [0] UTF8String  OPTIONAL,
        uri     [1] IA5String   OPTIONAL
   }

   Algorithm ::= SEQUENCE {
        algorithmIdentifier     AlgID,
        evaluations             SEQUENCE OF Evaluation,
        information         [0] SEQUENCE OF UTF8String  OPTIONAL,
        other               [1] Extension               OPTIONAL
   }

   Extension ::= SEQUENCE {
        extensionType           OBJECT IDENTIFIER,
        extension               ANY DEFINED BY extensionType
   }

   AlgID ::= SEQUENCE {
        name      UTF8String,
        oid   [0] SEQUENCE OF OBJECT IDENTIFIER,
        uri   [1] SEQUENCE OF IA5String          OPTIONAL
   }

   Evaluation ::= SEQUENCE {
        parameters           [0] SEQUENCE OF Parameter OPTIONAL,
```

```
        validity            [1] Validity,
        other               [2] Extension            OPTIONAL
    }

    Parameter ::= SEQUENCE {
        name       UTF8String,
        min    [0] INTEGER     OPTIONAL,
        max    [1] INTEGER     OPTIONAL,
        other  [2] Extension   OPTIONAL
    }

    Validity ::= SEQUENCE {
        start  [0] GeneralizedTime OPTIONAL,
        end    [1] GeneralizedTime OPTIONAL
    }

    END
```

Appendix D. ASN.1 Module in 1997 Syntax (Normative)

```
    ASN.1-Module

    DSSC {iso(1) identified-organization(3) dod(6)
             internet(1) security(5) mechanisms(5)
             ltans(11) id-mod(0) id-mod-dssc(7) id-mod-dssc-v1(1) }

    DEFINITIONS IMPLICIT TAGS ::=
    BEGIN

    -- EXPORT ALL --

    IMPORTS

    -- Import from RFC 5280 [RFC5280]
    -- Delete following import statement
    -- if "new" types are supported

    UTF8String FROM PKIX1Explicit88
                { iso(1) identified-organization(3) dod(6)
                internet(1) security(5) mechanisms(5) pkix(7)
                mod(0) pkix1-explicit(18) }

    -- Import from RFC 5652 [RFC5652]

    ContentInfo FROM CryptographicMessageSyntax2004
                { iso(1) member-body(2) us(840)
                rsadsi(113549) pkcs(1) pkcs-9(9)
                smime(16) modules(0) cms-2004(24)}
```

```
   ;

   SecuritySuitabilityPolicy ::= ContentInfo

   -- contentType is id-signedData as defined in [RFC5652]
   -- content is SignedData as defined in [RFC5652]
   -- eContentType within SignedData is id-ct-dssc
   -- eContent within SignedData is TBSPolicy

   id-ct-dssc  OBJECT IDENTIFIER ::= {
               iso(1) identified-organization(3) dod(6)
               internet(1) security(5) mechanisms(5)
               ltans(11) id-ct(1) id-ct-dssc-tbsPolicy(6) }

   TBSPolicy ::= SEQUENCE {
        version          INTEGER               DEFAULT   {v1(1)},
        language         UTF8String            DEFAULT   "en",
        policyName       PolicyName,
        publisher        Publisher,
        policyIssueDate  GeneralizedTime,
        nextUpdate       GeneralizedTime       OPTIONAL,
        usage            UTF8String            OPTIONAL,
        algorithms       SEQUENCE OF Algorithm
   }

   PolicyName ::= SEQUENCE {
        name  UTF8String,
        oid   OBJECT IDENTIFIER OPTIONAL,
        uri   IA5String         OPTIONAL
   }

   Publisher ::= SEQUENCE {
        name         UTF8String,
        address  [0] UTF8String  OPTIONAL,
        uri      [1] IA5String   OPTIONAL
   }

   Algorithm ::= SEQUENCE {
        algorithmIdentifier     AlgID,
        evaluations             SEQUENCE OF Evaluation,
        information         [0] SEQUENCE OF UTF8String  OPTIONAL,
        other               [1] Extension               OPTIONAL
   }

   Extension ::= SEQUENCE {
        extensionType  EXTENSION-TYPE.&id ({SupportedExtensions}),
        extension      EXTENSION-TYPE.&Type
                            ({SupportedExtensions}{@extensionType})
```

```
   }

   EXTENSION-TYPE ::= TYPE-IDENTIFIER

   SupportedExtensions EXTENSION-TYPE ::= {...}

   AlgID ::= SEQUENCE {
        name      UTF8String,
        oid   [0] SEQUENCE OF OBJECT IDENTIFIER,
        uri   [1] SEQUENCE OF IA5String          OPTIONAL
   }

   Evaluation ::= SEQUENCE {
        parameters           [0] SEQUENCE OF Parameter OPTIONAL,
        validity             [1] Validity,
        other                [2] Extension             OPTIONAL
   }

   Parameter ::= SEQUENCE {
        name       UTF8String,
        min    [0] INTEGER     OPTIONAL,
        max    [1] INTEGER     OPTIONAL,
        other  [2] Extension   OPTIONAL
   }

   Validity ::= SEQUENCE {
        start  [0] GeneralizedTime OPTIONAL,
        end    [1] GeneralizedTime OPTIONAL
   }

   END
```

Appendix E. Example

The following example shows a policy that may be used for signature verification. It contains hash algorithms, public key algorithms, and signature schemes. SHA-1 as well as RSA with modulus length of 1024 are examples for expired algorithms.

```
<SecuritySuitabilityPolicy xmlns="urn:ietf:params:xml:ns:dssc"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <PolicyName>
  <Name>Evaluation of cryptographic algorithms</Name>
 </PolicyName>
 <Publisher>
  <Name>Some Evaluation Authority</Name>
 </Publisher>
 <PolicyIssueDate>2009-01-01T00:00:00</PolicyIssueDate>
```

```
    <Usage>Digital signature verification</Usage>
    <Algorithm>
     <AlgorithmIdentifier>
      <Name>SHA-1</Name>
      <ObjectIdentifier>1.3.14.3.2.26</ObjectIdentifier>
     </AlgorithmIdentifier>
     <Evaluation>
      <Validity>
       <End>2008-06-30</End>
      </Validity>
     </Evaluation>
    </Algorithm>
    <Algorithm>
     <AlgorithmIdentifier>
      <Name>SHA-256</Name>
      <ObjectIdentifier>2.16.840.1.101.3.4.2.1</ObjectIdentifier>
     </AlgorithmIdentifier>
     <Evaluation>
      <Validity>
       <End>2014-12-31</End>
      </Validity>
     </Evaluation>
    </Algorithm>
    <Algorithm>
     <AlgorithmIdentifier>
      <Name>SHA-512</Name>
      <ObjectIdentifier>2.16.840.1.101.3.4.2.3</ObjectIdentifier>
     </AlgorithmIdentifier>
     <Evaluation>
      <Validity>
       <End>2014-12-31</End>
      </Validity>
     </Evaluation>
    </Algorithm>
    <Algorithm>
     <AlgorithmIdentifier>
      <Name>RSA</Name>
      <ObjectIdentifier>1.2.840.113549.1.1.1</ObjectIdentifier>
     </AlgorithmIdentifier>
     <Evaluation>
      <Parameter name="moduluslength">
       <Min>1024</Min>
      </Parameter>
      <Validity>
       <End>2008-03-31</End>
      </Validity>
     </Evaluation>
     <Evaluation>
```

```
      <Parameter name="moduluslength">
       <Min>2048</Min>
      </Parameter>
      <Validity>
       <End>2014-12-31</End>
      </Validity>
     </Evaluation>
    </Algorithm>
    <Algorithm>
     <AlgorithmIdentifier>
      <Name>DSA</Name>
      <ObjectIdentifier>1.2.840.10040.4.1</ObjectIdentifier>
     </AlgorithmIdentifier>
     <Evaluation>
      <Parameter name="plength">
       <Min>1024</Min>
      </Parameter>
      <Parameter name="qlength">
       <Min>160</Min>
      </Parameter>
      <Validity>
       <End>2007-12-31</End>
      </Validity>
     </Evaluation>
     <Evaluation>
      <Parameter name="plength">
       <Min>2048</Min>
      </Parameter>
      <Parameter name="qlength">
       <Min>224</Min>
      </Parameter>
      <Validity>
       <End>2014-12-31</End>
      </Validity>
     </Evaluation>
    </Algorithm>
    <Algorithm>
     <AlgorithmIdentifier>
      <Name>PKCS#1 v1.5 SHA-1 with RSA</Name>
      <ObjectIdentifier>1.2.840.113549.1.1.5</ObjectIdentifier>
     </AlgorithmIdentifier>
     <Evaluation>
      <Parameter name="moduluslength">
       <Min>1024</Min>
      </Parameter>
      <Validity>
       <End>2008-03-31</End>
      </Validity>
```

```
     </Evaluation>
     <Evaluation>
      <Parameter name="moduluslength">
       <Min>2048</Min>
      </Parameter>
      <Validity>
       <End>2008-06-30</End>
      </Validity>
     </Evaluation>
    </Algorithm>
    <Algorithm>
     <AlgorithmIdentifier>
      <Name>PKCS#1 v1.5 SHA-256 with RSA</Name>
      <ObjectIdentifier>1.2.840.113549.1.1.11</ObjectIdentifier>
     </AlgorithmIdentifier>
     <Evaluation>
      <Parameter name="moduluslength">
       <Min>1024</Min>
      </Parameter>
      <Validity>
       <End>2008-03-31</End>
      </Validity>
     </Evaluation>
     <Evaluation>
      <Parameter name="moduluslength">
       <Min>2048</Min>
      </Parameter>
      <Validity>
       <End>2014-12-31</End>
      </Validity>
     </Evaluation>
    </Algorithm>
    <Algorithm>
     <AlgorithmIdentifier>
      <Name>PKCS#1 v1.5 SHA-512 with RSA</Name>
      <ObjectIdentifier>1.2.840.113549.1.1.13</ObjectIdentifier>
     </AlgorithmIdentifier>
     <Evaluation>
      <Parameter name="moduluslength">
       <Min>1024</Min>
      </Parameter>
      <Validity>
       <End>2008-03-31</End>
      </Validity>
     </Evaluation>
     <Evaluation>
      <Parameter name="moduluslength">
       <Min>2048</Min>
```

```
      </Parameter>
      <Validity>
       <End>2014-12-31</End>
      </Validity>
     </Evaluation>
    </Algorithm>
    <Algorithm>
     <AlgorithmIdentifier>
      <Name>SHA-1 with DSA</Name>
      <ObjectIdentifier>1.2.840.10040.4.3</ObjectIdentifier>
     </AlgorithmIdentifier>
     <Evaluation>
      <Parameter name="plength">
       <Min>1024</Min>
      </Parameter>
      <Parameter name="qlength">
       <Min>160</Min>
      </Parameter>
      <Validity>
       <End>2007-12-31</End>
      </Validity>
     </Evaluation>
     <Evaluation>
      <Parameter name="plength">
       <Min>2048</Min>
      </Parameter>
      <Parameter name="qlength">
       <Min>224</Min>
      </Parameter>
      <Validity>
       <End>2008-06-30</End>
      </Validity>
     </Evaluation>
    </Algorithm>
   </SecuritySuitabilityPolicy>
```

Authors' Addresses

Thomas Kunz
Fraunhofer Institute for Secure Information Technology
Rheinstrasse 75
Darmstadt D-64295
Germany

EMail: thomas.kunz@sit.fraunhofer.de

Susanne Okunick
pawisda systems GmbH
Robert-Koch-Strasse 9
Weiterstadt D-64331
Germany

EMail: susanne.okunick@pawisda.de

Ulrich Pordesch
Fraunhofer Gesellschaft
Rheinstrasse 75
Darmstadt D-64295
Germany

EMail: ulrich.pordesch@zv.fraunhofer.de