               PKCS #10: Certification Request Syntax Specification
                              Version 1.7

Status of this Memo

Copyright Notice

Abstract

   This memo represents a republication of PKCS #10 v1.7 from RSA
   Laboratories' Public-Key Cryptography Standards (PKCS) series, and
   change control is retained within the PKCS process. The body of this
   document, except for the security considerations section, is taken
   directly from the PKCS #9 v2.0 or the PKCS #10 v1.7 document.

   This memo describes a syntax for certification requests.

Table of Contents

1. Introduction

   This document describes syntax for certification requests. A
   certification request consists of a distinguished name, a public key,
   and optionally a set of attributes, collectively signed by the entity
   requesting certification. Certification requests are sent to a
   certification authority, which transforms the request into an X.509
   [9] public-key certificate. (In what form the certification
   authority returns the newly signed certificate is outside the scope
   of this document. A PKCS #7 [2] message is one possibility.)

   The intention of including a set of attributes is twofold: to provide
   other information about a given entity , or a "challenge password" by
   which the entity may later request certificate revocation; and to
   provide attributes for inclusion in X.509 certificates. A non-
   exhaustive list of attributes is given in PKCS #9 [3].

   Certification authorities may also require non-electronic forms of
   request and may return non-electronic replies. It is expected that
   descriptions of such forms, which are outside the scope of this
   document, will be available from certification authorities.

   The preliminary intended application of this document is to support
   PKCS #7 cryptographic messages, but it is expected that other
   applications will be developed (see e.g. [4]).

2. Definitions and notation

 2.1 Definitions

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

   ALGORITHM   An information object class defined in X.509 to
                    describe objects composed of an algorithm (a unique
                    object identifier) and its parameters (any ASN.1
                    type). The values of objects in this class can be
                    represented by the ASN.1 type AlgorithmIdentifier{}.
                    ALGORITHM is defined as the "useful" information
                    object class TYPE-IDENTIFIER, specified in [11],
                    Annex A.

   AlgorithmIdentifier{}
                    A useful parameterized version of X.509 type
                    AlgorithmIdentifier is defined in this document.
                    This type tightly binds pairs of algorithm object
                    identifiers to their associated parameter types.
                    When referenced, the single parameter of
                    AlgorithmIdentifier{} specifies a constraint on the

                    pairs of values that may appear in that instance of
                    the type. The encoded values of
                    AlgorithmIdentifier{} are equivalent to those of type
                    AlgorithmIdentifier.

   ASN.1   Abstract Syntax Notation One, as defined in the ASN.1
                    standards ([10], [11], [12], and [13]).

   ATTRIBUTE   This class describes objects composed of an attribute
                    (a unique object identifier) and an associated set of
                    attribute values (any ASN.1 type). The values of
                    objects in this class can be represented by type
                    Attribute{}.

   Attribute{}   A useful parameterized version of X.501 [8] type
                    Attribute is defined in this document. This type
                    tightly binds pairs of attribute type object
                    identifiers to one or more attribute values types.
                    In the ASN.1 open type notation, an attribute type is
                    defined as ATTRIBUTE.&id and an attribute value as
                    ATTRIBUTE.&Type. When referenced, the single
                    parameter of Attribute{} specifies a constraint on
                    the pairs of values that may appear in an instance of
                    the type. The encoded values of Attribute{} are
                    equivalent to those of type Attribute.

   BER   Basic Encoding Rules for ASN.1, as defined in X.690
                    ([14]).

   Certificate   A type that binds a subject entity's distinguished
                    name to a public key with a digital signature. This
                    type is defined in X.509. This type also contains
                    the distinguished name of the certificate issuer (the
                    signer), an issuer-specific serial number, the
                    issuer's signature algorithm identifier, a validity
                    period, and an optional set of certificate
                    extensions.

   DER   Distinguished Encoding Rules for ASN.1, as defined in
                    X.690. DER is a subset of BER.

   Name   A type that uniquely identifies or "distinguishes"
                    objects in an X.500 [7] directory. This type is
                    defined in X.501. In an X.509 certificate, the type
                    identifies the certificate issuer and the certificate
                    subject, the entity whose public key is certified.

2.2 Notation

   No special notation is used in this document.

3. Overview

   A certification request consists of three parts: "certification
   request information," a signature algorithm identifier, and a digital
   signature on the certification request information.   The
   certification request information consists of the entity's
   distinguished name, the entity's public key, and a set of attributes
   providing other information about the entity.

   The process by which a certification request is constructed involves
   the following steps:

        1. A CertificationRequestInfo value containing a subject
           distinguished name, a subject public key, and optionally a
           set of attributes is constructed by an entity requesting
           certification.

        2. The CertificationRequestInfo value is signed with the subject
           entity's private key. (See Section 4.2.)

        3. The CertificationRequestInfo value, a signature algorithm
           identifier, and the entity's signature are collected together
           into a CertificationRequest value, defined below.

   A certification authority fulfills the request by authenticating the
   requesting entity and verifying the entity's signature, and, if the
   request is valid, constructing an X.509 certificate from the
   distinguished name and public key, the issuer name, and the
   certification authority's choice of serial number, validity period,
   and signature algorithm. If the certification request contains any
   PKCS #9 attributes, the certification authority may also use the
   values in these attributes as well as other information known to the
   certification authority to construct X.509 certificate extensions.

   In what form the certification authority returns the new certificate
   is outside the scope of this document. One possibility is a PKCS #7
   cryptographic message with content type signedData, following the
   degenerate case where there are no signers. The return message may
   include a certification path from the new certificate to the
   certification authority. It may also include other certificates such
   as cross-certificates that the certification authority considers
   helpful, and it may include certificate-revocation lists (CRLs).
   Another possibility is that the certification authority inserts the
   new certificate into a central database.

   Note 1 - An entity would typically send a certification request after
   generating a public-key/private-key pair, but may also do so after a
   change in the entity's distinguished name.

   Note 2 - The signature on the certification request prevents an
   entity from requesting a certificate with another party's public key.
   Such an attack would give the entity the minor ability to pretend to
   be the originator of any message signed by the other party. This
   attack is significant only if the entity does not know the message
   being signed and the signed part of the message does not identify the
   signer. The entity would still not be able to decrypt messages
   intended for the other party, of course.

   Note 3 - How the entity sends the certification request to a
   certification authority is outside the scope of this document. Both
   paper and electronic forms are possible.

   Note 4 - This document is not compatible with the certification
   request syntax for Privacy-Enhanced Mail, as described in RFC 1424
   [5]. The syntax here differs in three respects: It allows a set of
   attributes; it does not include issuer name, serial number, or
   validity period; and it does not require an "innocuous" message to be
   signed. This document is designed to minimize request size, an
   important feature for certification authorities accepting requests on
   paper.

4. Certification request syntax

   This section is divided into two parts. The first part describes the
   certification-request-information type CertificationRequestInfo, and
   the second part describes the top-level type CertificationRequest.

 4.1 CertificationRequestInfo

   Certification request information shall have ASN.1 type
   CertificationRequestInfo:

   CertificationRequestInfo ::= SEQUENCE {
        version   INTEGER { v1(0) } (v1,...),
        subject   Name,
        subjectPKInfo SubjectPublicKeyInfo{{ PKInfoAlgorithms }},
        attributes   [0] Attributes{{ CRIAttributes }}
   }

   SubjectPublicKeyInfo { ALGORITHM : IOSet} ::= SEQUENCE {
        algorithm   AlgorithmIdentifier {{IOSet}},
        subjectPublicKey BIT STRING
   }

```
   PKInfoAlgorithms ALGORITHM ::= {
       ... -- add any locally defined algorithms here -- }

   Attributes { ATTRIBUTE:IOSet } ::= SET OF Attribute{{ IOSet }}

   CRIAttributes  ATTRIBUTE  ::= {
       ... -- add any locally defined attributes here -- }

   Attribute { ATTRIBUTE:IOSet } ::= SEQUENCE {
       type   ATTRIBUTE.&id({IOSet}),
       values SET SIZE(1..MAX) OF ATTRIBUTE.&Type({IOSet}{@type})
   }
```

The components of type CertificationRequestInfo have the following
meanings:

> version is the version number, for compatibility with future
> revisions of this document. It shall be 0 for this version of
> the standard.

> subject is the distinguished name of the certificate subject
> (the entity whose public key is to be certified).

> subjectPublicKeyInfo contains information about the public key
> being certified. The information identifies the entity's
> public-key algorithm (and any associated parameters); examples
> of public-key algorithms include the rsaEncryption object
> identifier from PKCS #1 [1]. The information also includes a
> bit-string representation of the entity's public key. For the
> public-key algorithm just mentioned, the bit string contains
> the DER encoding of a value of PKCS #1 type RSAPublicKey. The
> values of type SubjectPublicKeyInfo{} allowed for
> subjectPKInfo are constrained to the values specified by the
> information object set PKInfoAlgorithms, which includes the
> extension marker (...). Definitions of specific algorithm
> objects are left to specifications that reference this
> document. Such specifications will be interoperable with
> their future versions if any additional algorithm objects are
> added after the extension marker.

> attributes is a collection of attributes providing additional
> information about the subject of the certificate. Some
> attribute types that might be useful here are defined in PKCS
> #9. An example is the challenge-password attribute, which
> specifies a password by which the entity may request
> certificate revocation. Another example is information to
> appear in X.509 certificate extensions (e.g. the
> extensionRequest attribute from PKCS #9). The values of type

          Attributes{} allowed for attributes are constrained to the
          values specified by the information object set CRIAttributes.
          Definitions of specific attribute objects are left to
          specifications that reference this document. Such
          specifications will be interoperable with their future
          versions if any additional attribute objects are added after
          the extension marker.

 4.2 CertificationRequest

   A certification request shall have ASN.1 type CertificationRequest:

   CertificationRequest ::= SEQUENCE {
        certificationRequestInfo CertificationRequestInfo,
        signatureAlgorithm AlgorithmIdentifier{{ SignatureAlgorithms }},
        signature   BIT STRING
   }

   AlgorithmIdentifier {ALGORITHM:IOSet } ::= SEQUENCE {
        algorithm   ALGORITHM.&id({IOSet}),
        parameters   ALGORITHM.&Type({IOSet}{@algorithm}) OPTIONAL
   }

   SignatureAlgorithms ALGORITHM ::= {
        ... -- add any locally defined algorithms here -- }

   The components of type CertificationRequest have the following
   meanings:

        certificateRequestInfo is the "certification request
          information." It is the value being signed.

        signatureAlgorithm identifies the signature algorithm (and any
          associated parameters) under which the certification-request
          information is signed. For example, a specification might
          include an ALGORITHM object for PKCS #1's
          md5WithRSAEncryption in the information object set
          SignatureAlgorithms:

        SignatureAlgorithms ALGORITHM ::= {
            ...,
            { NULL IDENTIFIED BY md5WithRSAEncryption }
        }

     signature is the result of signing the certification request
          information with the certification request subject's private
          key.

The signature process consists of two steps:

    1. The value of the certificationRequestInfo component is DER
       encoded, yielding an octet string.

    2. The result of step 1 is signed with the certification request
       subject's private key under the specified signature
       algorithm, yielding a bit string, the signature.

Note - An equivalent syntax for CertificationRequest could be
written:

```
CertificationRequest ::= SIGNED { EncodedCertificationRequestInfo }
     (CONSTRAINED BY { -- Verify or sign encoded
      -- CertificationRequestInfo -- })

EncodedCertificationRequestInfo ::=
     TYPE-IDENTIFIER.&Type(CertificationRequestInfo)

SIGNED { ToBeSigned } ::= SEQUENCE {
     toBeSigned ToBeSigned,
     algorithm  AlgorithmIdentifier { {SignatureAlgorithms} },
     signature  BIT STRING
}
```

5. Security Considerations

Security issues are discussed throughout this memo.

6. Authors' Addresses

Magnus Nystrom
RSA Security
Box 10704
S-121 29 Stockholm
Sweden

EMail: magnus@rsasecurity.com

Burt Kaliski
RSA Security
20 Crosby Drive
Bedford, MA 01730 USA

EMail: bkaliski@rsasecurity.com

APPENDICES

A. ASN.1 Module

   This appendix includes all of the ASN.1 type and value definitions
   contained in this document in the form of the ASN.1 module PKCS-10.

   PKCS-10 {iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
   pkcs-10(10) modules(1) pkcs-10(1)}

   DEFINITIONS IMPLICIT TAGS ::=

   BEGIN

   -- EXPORTS All --

   -- All types and values defined in this module are exported for use
   -- in other ASN.1 modules.

   IMPORTS

   informationFramework, authenticationFramework
       FROM UsefulDefinitions {joint-iso-itu-t(2) ds(5) module(1)
       usefulDefinitions(0) 3}

   ATTRIBUTE, Name
       FROM InformationFramework informationFramework

   ALGORITHM
       FROM AuthenticationFramework authenticationFramework;

   -- Certificate requests
   CertificationRequestInfo ::= SEQUENCE {
       version   INTEGER { v1(0) } (v1,...),
       subject   Name,
       subjectPKInfo SubjectPublicKeyInfo{{ PKInfoAlgorithms }},
       attributes   [0] Attributes{{ CRIAttributes }}
   }

   SubjectPublicKeyInfo {ALGORITHM: IOSet} ::= SEQUENCE {
       algorithm   AlgorithmIdentifier {{IOSet}},
       subjectPublicKey BIT STRING
   }

   PKInfoAlgorithms ALGORITHM ::= {
       ... -- add any locally defined algorithms here -- }

   Attributes { ATTRIBUTE:IOSet } ::= SET OF Attribute{{ IOSet }}

```
    CRIAttributes  ATTRIBUTE  ::= {
         ... -- add any locally defined attributes here -- }

    Attribute { ATTRIBUTE:IOSet } ::= SEQUENCE {
         type   ATTRIBUTE.&id({IOSet}),
         values SET SIZE(1..MAX) OF ATTRIBUTE.&Type({IOSet}{@type})
    }

    CertificationRequest ::= SEQUENCE {
         certificationRequestInfo CertificationRequestInfo,
         signatureAlgorithm AlgorithmIdentifier{{ SignatureAlgorithms }},
         signature          BIT STRING
    }

    AlgorithmIdentifier {ALGORITHM:IOSet } ::= SEQUENCE {
         algorithm  ALGORITHM.&id({IOSet}),
         parameters ALGORITHM.&Type({IOSet}{@algorithm}) OPTIONAL
    }

    SignatureAlgorithms ALGORITHM ::= {
         ... -- add any locally defined algorithms here -- }

    END
```

B. Intellectual property considerations

   RSA Security makes no patent claims on the general constructions
   described in this document, although specific underlying techniques
   may be covered.

   License to copy this document is granted provided that it is
   identified as "RSA Security Inc. Public-Key Cryptography Standards
   (PKCS)" in all material mentioning or referencing this document.

   RSA Security makes no representations regarding intellectual property
   claims by other parties. Such determination is the responsibility of
   the user.

C. Revision history

   Version 1.0

        Version 1.0 was the previous version of this document (also
        published as "version 1.5" in [6]).

Version 1.7

        This version incorporates several editorial changes, including
        updates to the references, and changes to ASN.1 type
        definitions. The following substantive changes have been made:

        - This version refers to X.680-X.690, the current international
          standards for ASN.1 and its encoding rules. All references
          to X.208 and X.209 have been eliminated.

        - The X.690 standard requires that the encoded values of SET OF
          components be sorted in ascending order under DER.
          Regardless of this, applications should not rely on the
          ordering of attribute components.

        - All references to PKCS #6 Extended-Certificate Syntax
          Standard have been removed. With the addition of extensions
          to X.509 version 3 certificates, RSA Laboratories is
          withdrawing support for PKCS #6.

    Note - The reason for using version 1.7 for this document is to avoid
    confusion with [6], which is named version 1.5, and an unsupported
    PKCS #10 version named Version 1.6.

D. References

   [1] RSA Laboratories. PKCS #1: RSA Encryption Standard. Version 2.0,
         October 1998.

   [2] RSA Laboratories. PKCS #7: Cryptographic Message Syntax
         Standard. Version 1.5, November 1993.

   [3] RSA Laboratories. PKCS #9: Selected Attribute Types. Version
         2.0, February 2000.

   [4] Adams, C. and S. Farrell, "Internet X.509 Public Key
         Infrastructure - Certificate Management Protocols", RFC 2510,
         March 1999.

   [5] Kaliski, B., "Privacy Enhancement for Internet Electronic Mail:
         Part IV: Key Certification and Related Services", RFC 1424,
         February 1993.

   [6] Kaliski, B., "PKCS #10: Certification Request Syntax Version
         1.5", RFC 2314, March 1998.

   [7] ITU-T Recommendation X.500 (1997) | ISO/IEC 9594-1:1998,
        Information technology - Open Systems Interconnection - The
        Directory: Overview of concepts, models and services.

   [8] ITU-T Recommendation X.501 (1993) | ISO/IEC 9594-2:1995,
        Information technology - Open Systems Interconnection - The
        Directory: Models.

   [9] ITU-T Recommendation X.509 (1997) | ISO/IEC 9594-8:1998,
        Information technology - Open Systems Interconnection -The
        Directory: Authentication framework.

   [10] ITU-T Recommendation X.680 (1997) | ISO/IEC 8824-1:1998,
        Information Technology - Abstract Syntax Notation One (ASN.1):
        Specification of Basic Notation.

   [11] ITU-T Recommendation X.681 (1997) | ISO/IEC 8824-2:1998,
        Information Technology - Abstract Syntax Notation One (ASN.1):
        Information Object Specification.

   [12] ITU-T Recommendation X.682 (1997) | ISO/IEC 8824-3:1998,
        Information Technology - Abstract Syntax Notation One (ASN.1):
        Constraint Specification.

   [13] ITU-T Recommendation X.683 (1997) | ISO/IEC 8824-4:1998,
        Information Technology - Abstract Syntax Notation One (ASN.1):
        Parameterization of ASN.1 Specifications.

   [14] ITU-T Recommendation X.690 (1997) | ISO/IEC 8825-1:1998,
        Information Technology - ASN.1 Encoding Rules: Specification of
        Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and
        Distinguished Encoding Rules (DER).

E. Contact Information & About PKCS

   The Public-Key Cryptography Standards are specifications produced by
   RSA Laboratories in cooperation with secure systems developers
   worldwide for the purpose of accelerating the deployment of public-
   key cryptography. First published in 1991 as a result of meetings
   with a small group of early adopters of public-key technology, the
   PKCS documents have become widely referenced and implemented.
   Contributions from the PKCS series have become part of many formal
   and de facto standards, including ANSI X9 documents, PKIX, SET,
   S/MIME, and SSL.

Further development of PKCS occurs through mailing list discussions
and occasional workshops, and suggestions for improvement are
welcome. For more information, contact:

        PKCS Editor
        RSA Laboratories
        20 Crosby Drive
        Bedford, MA 01730 USA
        pkcs-editor@rsasecurity.com
        http://www.rsasecurity.com/rsalabs/pkcs

Acknowledgement