PKCS #9 v2.0: Selected Object Classes and Attribute Types (Final Draft)

RSA Laboratories

4 February, 2000

Editor's note*: This is the final draft of PKCS #9 v2.0, which is available for a 14-day public review period. Please send comments and suggestions, both technical and editorial, to* pkcs-editor@rsasecurity.com *or* pkcs-tng@rsasecurity.com

Table of Contents

000–000000–000–000

1. Introduction

This document defines two new auxiliary object classes, **pkcsEntity** and **naturalPerson**, and selected attribute types for use with these classes. It also defines some attribute types for use in conjunction with PKCS #7 [14] (and S/MIME CMS [3]) digitally signed messages, PKCS #10 [16] certificate-signing requests, PKCS #12 [17] personal information exchanges and PKCS #15 [18] cryptographic tokens. Matching rules for use with these attributes are also defined, whenever necessary.

2. Definitions, notation and document conventions

2.1 Definitions

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

ASN.1 Abstract Syntax Notation One, as defined in [5].

Attributes An ASN.1 type that specifies a set of attributes. Each attribute contains an attribute type (specified by object identifier) and one or more attribute values. Some attribute types are restricted in their definition to have a single value; others may have multiple values. This type is defined in [7].

CertificationRequestInfo An ASN.1 type that specifies a subject name, a public key, and a set of attributes. This type is defined in [16].

ContentInfo An ASN.1 type that specifies content exchanged between entities. The **contentType** field, which has type **OBJECT IDENTIFIER**, specifies the content type, and the **content** field, whose type is defined by the **contentType** field, contains the content value. This type is defined in [14] and [3].

PrivateKeyInfo A type that specifies a private key and a set of extended attributes. This type and the associated **EncryptedPrivateKeyInfo** type are defined in [15].

SignerInfo A type that specifies per-signer information in the signed-data content type, including a set of attributes authenticated by the signer, and a set of attributes not authenticated by the signer. This type is defined in [14] and [3].

DER Distinguished Encoding Rules for ASN.1, as defined in [6].

UCS Universal Multiple-Octet Coded Character Set, as defined in [11].

UTF8String	UCS Transformation Format encoded string. The UTF-8 encoding is defined in [11].

2.2 Notation and document conventions

In this document, all ASN.1 types and values are written in **bold Helvetica**. Attribute type and object class definitions are written in the ASN.1 *value notation* defined in [5]. Appendix B contains most of these definitions written in the augmented BNF notation defined in [2] as well. This has been done in an attempt to simplify the task of integrating this work into LDAP [22] development environments.

The keywords "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].

3. Overview

This document specifies two new auxiliary object classes, **pkcsEntity** and **naturalPerson**, and some new attribute types and matching rules. All ASN.1 object classes, attributes, matching rules and types are exported for use in other environments.

Attribute types defined in this document that are useful in conjunction with storage of PKCS-related data and the **pkcsEntity** object class includes PKCS #12 **PFX** PDUs, PKCS #15 tokens and encrypted private keys.

Attribute types defined in this document that are useful in conjunction with PKCS #10 certificate requests and the **naturalPerson** object class includes electronic-mail address, pseudonym, unstructured name, and unstructured address.

Attribute types defined in this document that are useful in PKCS #7 digitally signed messages are content type, message digest, signing time, sequence number, random nonce and countersignature. The attributes would be used in the **authenticatedAttributes** and **unauthenticatedAttributes** fields of a **SignerInfo** or an **AuthenticatedData** ([3]) value.

Attribute types that are useful especially in PKCS #10 certification requests are the challenge password and the extension-request attribute. The attributes would be used in the **attributes** field of a **CertificationRequestInfo** value.

Note – The attributes types (from [8]) in Table 1, and probably several others, might also be helpful in PKCS #10, PKCS #12 and PKCS #15-aware applications.

businessCategory	**preferredDeliveryMethod**
commonName	**presentationAddress**
countryName	**registeredAddress**
description	**roleOccupant**
destinationIndicator	**serialNumber**

facsimileTelephoneNumber	**stateOrProvinceName**
iSDNAddress	**streetAddress**
localityName	**supportedApplicationContext**
member	**surname**
objectClass	**telephoneNumber**
organizationName	**teletexTerminalIdentifier**
physicalDeliveryOfficeName	**telexNumber**
postalAddress	**title**
postalCode	**x121Address**
postOfficeBox	

Table 1: ISO/IEC 9594-6 attribute types useful in PKCS documents

4. Auxiliary object classes

This document defines two new auxiliary object classes: **pkcsEntity** and **naturalPerson**.

4.1 The pkcsEntity auxiliary object class

The **pkcsEntity** object class is a general-purpose auxiliary object class that is intended to hold attributes about PKCS-related entities. It has been designed for use within directory services based on the LDAP protocol [22] and the X.500 family of protocols, where support for PKCS-defined attributes is considered useful.

```
pkcsEntity OBJECT-CLASS     ::=       {
        SUBCLASS OF { top }
        KIND auxiliary
        MAY CONTAIN { PKCS9AttributeSet }
        ID pkcs-9-oc-pkcsEntity
}

PKCS9AttributeSet ATTRIBUTE ::= {
        userPKCS12 |
        pKCS15Token |
        encryptedPrivateKeyInfo,
        ... -- For future extensions
}
```

Attributes in the **PKCS9AttributeSet** are defined in Section 5.

4.2 The naturalPerson auxiliary object class

The **naturalPerson** object class is a general-purpose auxiliary object class that is intended to hold attributes about human beings. It has been designed for use within directory services based on the LDAP protocol [22] and the X.500 family of protocols, where support for these attributes is considered useful.

```
naturalPerson OBJECT-CLASS          ::=       {
```

```
        SUBCLASS OF { top }
        KIND auxilary
        MAY CONTAIN { NaturalPersonAttributeSet }
        ID pkcs-9-oc-naturalPerson
}

NaturalPersonAttributeSet ATTRIBUTE ::= {
        emailAddress |
        unstructuredName |
        unstructuredAddress |
        dateOfBirth |
        placeOfBirth |
        gender |
        countryOfCitizenship |
        countryOfResidence |
        pseudonym |
        serialNumber,
        ... -- For future extensions
}
```

Attributes in the **NaturalPersonAttributeSet** are defined in Section 5.

5. Selected attribute types

5.1 Attribute types for use with the "pkcsEntity" object class

5.1.1 PKCS #12 token

PKCS #12 provides a format for exchange of personal identity information. When such information is stored in a directory service, the **userPKCS12** attribute should be used.

```
userPKCS12 ATTRIBUTE ::= {
        WITH SYNTAX PFX
        ID pkcs-9-at-userPKCS12
}
```

This type was originally defined in [20].

5.1.2 PKCS #15 token

PKCS #15 provides a format for cryptographic tokens. When software variants of such tokens are stored in a directory service, the **pKCS15Token** attribute should be used.

```
pKCS15Token ATTRIBUTE ::= {
        WITH SYNTAX PKCS15Token
        ID pkcs-9-at-pkcs15Token
}
```

5.1.3 PKCS #8 encrypted private key information

PKCS #8 provides a format for encrypted private keys. When such information is stored in a directory service, the **encryptedPrivateKeyInfo** attribute should be used.

```
encryptedPrivateKeyInfo ATTRIBUTE ::= {
        WITH SYNTAX EncryptedPrivateKeyInfo
        ID pkcs-9-at-encryptedPrivateKeyInfo
}
```

5.2 Attribute types for use with the "naturalPerson" object class[1]

5.2.1 Electronic-mail address

The **emailAddress** attribute type specifies the electronic-mail address or addresses of a subject as an unstructured ASCII string. The interpretation of electronic-mail addresses is intended to be specified by certificate issuers etc.; no particular interpretation is required.

```
emailAddress ATTRIBUTE ::= {
        WITH SYNTAX IA5String (SIZE(1..pkcs-9-ub-emailAddress))
        EQUALITY MATCHING RULE pkcs9CaseIgnoreMatch
        ID pkcs-9-at-emailAdress
}
```

An electronic-mail address attribute can have multiple attribute values. When comparing two email addresses, case is irrelevant. The **pkcs9CaseIgnoreMatch** is defined in Section 6.

Note – It is likely that other standards bodies overseeing electronic-mail systems will, or have, registered electronic-mail address attribute types specific to their system. The electronic-mail address attribute type defined here was intended as a short-term substitute for those specific attribute types, but is included here for backwards-compatibility reasons.

5.2.2 Unstructured name

The **unstructuredName** attribute type specifies the name or names of a subject as an unstructured ASCII string. The interpretation of unstructured names is intended to be specified by certificate issuers etc.; no particular interpretation is required.

```
unstructuredName ATTRIBUTE ::= {
```

[1] Attributes **pseudonym**, **dateOfBirth**, **placeOfBirth**, **gender**, **countryOfResidence**, **countryOfResidence** has been defined in collaboration with the editors of the IETF PKIX' "Qualified Certificates" [19] document.

```
    WITH SYNTAX PKCS9String {pkcs-9-ub-unstructuredName}
    EQUALITY MATCHING RULE pkcs9CaseIgnoreMatch
    ID pkcs-9-at-unstructuredName
}

PKCS9String { INTEGER : maxSize} ::= CHOICE {
    ia5String IA5String (SIZE(1..maxSize)),
    directoryString DirectoryString {maxSize}
}
```

An unstructured-name attribute can have multiple attribute values. When comparing two unstructured names, case is irrelevant.

The **PKCS9String** type is defined as a choice of **IA5String** and **DirectoryString**. Applications SHOULD use the **IA5String** type when generating attribute values in accordance with this version of this document, unless internationalization issues makes this impossible. In that case, the **UTF8String** alternative of the **DirectoryString** alternative is the preferred choice. PKCS #9-attribute processing systems MUST be able to recognize and process all string types in **PKCS9String** values.

Note – Version 1.2 of this document defined **unstructuredName** as having the syntax **IA5String**, but did contain a note explaining that this might be changed to a **CHOICE** of different string types in future versions. To better accommodate international names, this type has been extended to also include a directory string in this version of this document. Since [21] does not support a directory string type containing **IA5String**s, a separate syntax object identifier has been defined (see [21] and Appendix B).

5.2.3 Unstructured address

The **unstructuredAddress** attribute type specifies the address or addresses of a subject as an unstructured directory string. The interpretation of unstructured addresses is intended to be specified by certificate issuers etc; no particular interpretation is required. A likely interpretation is as an alternative to the **postalAddress** attribute type defined in [8].

```
unstructuredAddress ATTRIBUTE ::= {
    WITH SYNTAX DirectoryString {pkcs-9-ub-unstructuredAddress}
    EQUALITY MATCHING RULE caseIgnoreMatch
    ID pkcs-9-at-unstructuredAddress
}
```

An unstructured-address attribute can have multiple attribute values. The **caseIgnoreMatch** matching rule is defined in [8].

Note 1 – It is recommended to use the ASN.1 type **TeletexString**'s new-line character (hexadecimal code `0d`) as a line separator in multi-line addresses.

Note 2 – Previous versions of this document defined **unstructuredAddress** as having the following syntax:

```
CHOICE {
        teletexString TeletexString,
        printableString PrintableString,
}
```

But also mentioned the possibility of a future definition as follows:

```
CHOICE {
        teletexString TeletexString,
        printableString PrintableString,
        universalString UniversalString
}
```

In this version of this document, the X.520 type **DirectoryString** has been used in order to be more aligned with international standards and current practice. When generating attribute values in accordance with this version of this document, applications SHOULD use the **PrintableString** alternative unless internationalization issues makes this impossible. In those cases, the **UTF8String** alternative SHOULD be used. PKCS #9-attribute processing systems MUST be able to recognize and process all string types in **DirectoryString** values.

5.2.4 Date of birth

The **dateOfBirth** attribute specifies the date of birth for the subject it is associated with.

```
dateOfBirth ATTRIBUTE ::= {
        WITH SYNTAX GeneralizedTime
        EQUALITY MATCHING RULE generalizedTimeMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-dateOfBirth
}
```

dateOfBirth attributes must be single-valued. The **generalizedTimeMatch** matching rule is defined in [8].

5.2.5 Place of birth

The **placeOfBirth** attribute specifies the place of birth for the subject it is associated with.

```
placeOfBirth ATTRIBUTE ::= {
        WITH SYNTAX DirectoryString {pkcs-9-ub-placeOfBirth}
        EQUALITY MATCHING RULE caseExactMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-placeOfBirth
}
```

placeOfBirth attributes must be single-valued. The **caseExactMatch** matching rule is defined in [8].

5.2.6 Gender

The **gender** attribute specifies the gender of the subject it is associated with.

```
gender ATTRIBUTE ::= {
        WITH SYNTAX PrintableString (SIZE(1) ^ FROM (“M” | ”F” | “m” | “f”))
        EQUALITY MATCHING RULE caseIgnoreMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-gender
}
```

The letter “M” (or “m”) represents “male” and the letter “F” (or “f”) represents “female”. **gender** attributes must be single-valued.

5.2.7 Country of citizenship

The **countryOfCitizenship** attribute specifies the (claimed) countries of citizenship for the subject it is associated with. It SHALL be a 2-letter acronym of a country in accordance with [4].

```
countryOfCitizenship ATTRIBUTE ::= {
        WITH SYNTAX PrintableString (SIZE(2) ^ CONSTRAINED BY {
        -- Must be a two-letter country acronym in accordance with ISO/IEC 3166 --})
        EQUALITY MATCHING RULE caseIgnoreMatch
        ID pkcs-9-at-countryOfCitizenship
}
```

Attributes of this type need not be single-valued.

5.2.8 Country of residence

The **countryOfResidence** attribute specifies the (claimed) country of residence for the subject is associated with. It SHALL be a 2-letter acronym of a country in accordance with [4].

```
countryOfResidence ATTRIBUTE ::= {
        WITH SYNTAX PrintableString (SIZE(2) ^ CONSTRAINED BY {
        -- Must be a two-letter country acronym in accordance with ISO/IEC 3166 --})
        EQUALITY MATCHING RULE caseIgnoreMatch
        ID pkcs-9-at-countryOfResidence
}
```

Attributes of this type need not be single-valued, since it is possible to be a resident of several countries.

5.2.9 Pseudonym

The **pseudonym** attribute type shall contain a pseudonym of a subject. The exact interpretation of pseudonyms is intended to be specified by certificate issuers etc.; no particular interpretation is required.

```
pseudonym ATTRIBUTE ::= {
        WITH SYNTAX DirectoryString {pkcs-9-ub-pseudonym}
        EQUALITY MATCHING RULE caseExactMatch
        ID id-at-pseudonym
}
```

Note – The **pseudonym** attribute has received an object identifier in the joint-iso-itu-t object identifier tree.

The **caseExactMatch** matching rule is defined in [8].

5.2.10 Serial number

The **serialNumber** attribute is defined in [8].

5.3 Attribute types for use in PKCS #7 data

5.3.1 Content type

The **contentType** attribute type specifies the content type of the **ContentInfo** value being signed in PKCS #7 (or S/MIME CMS) digitally signed data. In such data, the **contentType** attribute type is required if there are any PKCS #7 authenticated attributes.

```
contentType ATTRIBUTE ::= {
        WITH SYNTAX ContentType
        EQUALITY MATCHING RULE objectIdentifierMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-contentType
}

ContentType ::= OBJECT IDENTIFIER
```

As indicated, content-type attributes must have a single attribute value. For two content-type values to match, their octet string representation must be of equal length and corresponding octets identical. The **objectIdentifierMatch** matching rule is defined in [7].

Note – This attribute type is described in [3] as well.

5.3.2 Message digest

The **messageDigest** attribute type specifies the message digest of the contents octets of the DER-encoding of the **content** field of the **ContentInfo** value being signed in PKCS #7 digitally signed data, where the message digest is computed under the signer's message digest algorithm. The message-digest attribute type is required in these cases if there are any PKCS #7 authenticated attributes present.

```
messageDigest ATTRIBUTE ::= {
        WITH SYNTAX MessageDigest
        EQUALITY MATCHING RULE octetStringMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-messageDigest
}

MessageDigest ::= OCTET STRING
```

As indicated, a message-digest attribute must have a single attribute value. For two **messageDigest** values to match, their octet string representation must be of equal length and corresponding octets identical. The **octetStringMatch** matching rule is defined in [8].

Note – This attribute is described in [3] as well.

5.3.3 Signing time

The **signingTime** attribute type is intended for PKCS #7 digitally signed data. It specifies the time at which the signer (purportedly) performed the signing process.

```
signingTime ATTRIBUTE ::= {
        WITH SYNTAX SigningTime
        EQUALITY MATCHING RULE signingTimeMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-signingTime
}

SigningTime ::= Time -- imported from ISO/IEC 9594-8
```

A signing-time attribute must have a single attribute value

The **signingTimeMatch** matching rule (defined in Section 6.1) returns **TRUE** if an attribute value represents the same time as a presented value.

Quoting from [3]:

> "Dates between 1 January 1950 and 31 December 2049 (inclusive) MUST be encoded as **UTCTime**. Any dates with year values before 1950 or after 2049 MUST be encoded as **GeneralizedTime**. [Further,] **UTCTime** values MUST be expressed in Greenwich Mean Time (Zulu) and MUST include seconds (i.e., times are YYMMDDHHMMSSZ), even where the number of seconds is zero. Midnight

(GMT) must be represented as "YYMMDD000000Z". Century information is implicit, and the century shall be determined as follows:

- Where YY is greater than or equal to 50, the year shall be interpreted as 19YY; and

- Where YY is less than 50, the year shall be interpreted as 20YY.

GeneralizedTime values shall be expressed in Greenwich Mean Time (Zulu) and must include seconds (i.e., times are YYYYMMDDHHMMSSZ), even where the number of seconds is zero. **GeneralizedTime** values must not include fractional seconds."

Note 1 – The definition of SigningTime matches the definition of Time specified in [10].

Note 2 – No requirement is imposed concerning the correctness of the signing time, and acceptance of a purported signing time is a matter of a recipient's discretion. It is expected, however, that some signers, such as time-stamp servers, will be trusted implicitly.

5.3.4 Random nonce

The **randomNonce** attribute type is intended for PKCS #7 digitally signed data. It may be used by a signer unable (or unwilling) to specify the time at which the signing process was performed. Used in a correct manner, it will make it possible for the signer to protect against certain attacks, i.e. replay attacks.

```
randomNonce ATTRIBUTE ::= {
        WITH SYNTAX RandomNonce
        EQUALITY MATCHING RULE octetStringMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-randomNonce
}

RandomNonce ::= OCTET STRING (SIZE(4..MAX)) -- At least four bytes long
```

A random nonce attribute must have a single attribute value.

5.3.5 Sequence number

The **sequenceNumber** attribute type is intended for PKCS #7 digitally signed data. A signer wishing to associate a sequence number to all signature operations (much like a physical checkbook) may use it as an alternative to the **randomNonce** attribute. Used in a correct manner, it will make it possible for the signer to protect against certain attacks, i.e. replay attacks.

```
sequenceNumber ATTRIBUTE ::= {
        WITH SYNTAX SequenceNumber
```

```
    EQUALITY MATCHING RULE integerMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-sequenceNumber
}

SequenceNumber ::= INTEGER (1..MAX)
```

A sequence number attribute must have a single attribute value.

The **integerMatch** matching rule is defined in [8].

5.3.6 Countersignature

The **counterSignature** attribute type specifies one or more signatures on the content octets of the DER encoding of the **encryptedDigest** field of a **SignerInfo** value in PKCS #7 digitally signed data. Thus, the countersignature attribute type countersigns (signs in serial) another signature. The countersignature attribute must be an unauthenticated PKCS #7 attribute; it cannot be an authenticated attribute.

```
counterSignature ATTRIBUTE ::= {
    WITH SYNTAX SignerInfo
    ID pkcs-9-at-counterSignature
}
```

Countersignature values have the same meaning as **SignerInfo** values for ordinary signatures (see Section 9 of [14] and Section 5.3 of [3]), except that:

1. The **authenticatedAttributes** field must contain a **messageDigest** attribute if it contains any other attributes, but need not contain a **contentType** attribute, as there is no content type for countersignatures; and

2. The input to the message-digesting process is the content octets of the DER encoding of the **signatureValue** field of the **SignerInfo** value with which the attribute is associated.

A countersignature attribute can have multiple attribute values.

Note 1 – The fact that a countersignature is computed on a signature (encrypted digest) means that the countersigning process need not know the original content input to the signing process. This has advantages both in efficiency and in confidentiality.

Note 2 – A countersignature, since it has type **SignerInfo**, can itself contain a countersignature attribute. Thus it is possible to construct arbitrarily long series of countersignatures.

5.4 Attribute types for use with PKCS #10 certificate requests

5.4.1 Challenge password

The **challengePassword** attribute type specifies a password by which an entity may request certificate revocation. The interpretation of challenge passwords is intended to be specified by certificate issuers etc; no particular interpretation is required

```
challengePassword ATTRIBUTE ::= {
        WITH SYNTAX PKCS9String {pkcs-9-ub-challengePassword}
        EQUALITY MATCHING RULE pkcs9CaseExactMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-challengePassword
}
```

A challenge-password attribute must have a single attribute value.

ChallengePassword attribute values generated in accordance with this version of this document SHOULD use the **IA5String** encoding whenever possible. If internationalization issues make this impossible, the **UTF8String** alternative SHOULD be used. PKCS #9-attribute processing systems MUST be able to recognize and process all string types in **PKCS9String** values.

The **pkcs9CaseExactMatch** matching rule is defined in Section 6.

Note – Version 1.2 of this document defined **challengePassword** as having the syntax **IA5String**, but did contain a note explaining that this might be changed to a **CHOICE** of different string types in the future.

5.4.2 Extension request[2]

The **extensionRequest** attribute type may be used to carry information about certificate extensions the requester wishes to be included in a certificate.

```
extensionRequest ATTRIBUTE ::= {
     WITH SYNTAX ExtensionRequest
     SINGLE VALUE TRUE
     ID pkcs-9-at-extensionRequest
}

ExtensionRequest ::= Extensions
```

The **Extensions** type is imported from [10].

---

[2] The extension request attribute was originally used in RSA Data Security's TIPEM™ product.

5.4.3 Extended-certificate attributes (deprecated)

The **extendedCertificateAttributes** attribute type specified a set of attributes for a PKCS #6 [13] *extended certificate* in a PKCS #10 certification request (the value of the extended certificate-attributes attribute would become the extension in the requested PKCS #6 extended certificate). Since the status of PKCS #6 is historic after the introduction of X.509 v3 certificates [10], the use of this attribute is deprecated.

```
extendedCertificateAttributes ATTRIBUTE ::= {
        WITH SYNTAX SET OF Attribute
        SINGLE VALUE TRUE
        ID pkcs-9-at-extendedCertificateAttributes
}
```

An extended certificate attributes attribute must have a single attribute value (that value is a set, which itself may contain multiple values, but there must be only one set).

5.5 Attributes for use in PKCS #12 "PFX" PDUs or PKCS #15 tokens

5.5.1 Friendly name

The **friendlyName** attribute type specifies a user-friendly name of the object it belongs to. It is referenced in [17].

```
friendlyName ATTRIBUTE ::= {
        WITH SYNTAX BMPString (SIZE(1..pkcs-9-ub-friendlyName))
        EQUALITY MATCHING RULE caseIgnoreMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-friendlyName
}
```

As indicated, **friendlyName** attributes must have a single attribute value.

5.5.2 Local key identifier

The **localKeyId** attribute type specifies an identifier for a particular key. It is only to be used locally in applications. This attribute is referenced in [17].

```
localKeyId ATTRIBUTE ::= {
        WITH SYNTAX OCTET STRING
        EQUALITY MATCHING RULE octetStringMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-localKeyId
}
```

As indicated, **localKeyId** attributes must have a single attribute value. For two **localKeyId** values to match, their octet string representation must be of equal length and corresponding octets identical.

5.6 Attributes defined in S/MIME

S/MIME (c.f. [12]) defines some attributes and object identifiers in the PKCS #9 object identifier tree. For completeness, they are mentioned here.

5.6.1 Signing description[3]

The **signingDescription** attribute is intended to provide a short synopsis of a message that can be used to present a user with an additional confirmation step before committing to a cryptographic operation. In most cases, the replication of the "Subject:" line from the header of a message should be sufficient and is recommended.

```
signingDescription ATTRIBUTE ::= {
        WITH SYNTAX DirectoryString {pkcs-9-ub-signingDescription}
        EQUALITY MATCHING RULE caseIgnoreMatch
        SINGLE VALUE TRUE
        ID pkcs-9-at-signingDescription
}
```

5.6.2 S/MIME capabilities

The syntax and semantics of the **smimeCapabilities** attribute is defined in [12]. It is included here for the sake of completeness.

```
smimeCapabilities ATTRIBUTE ::= {
        WITH SYNTAX SMIMECapabilities
        SINGLE VALUE
        ID pkcs-9-at-smimeCapabilities
}

SMIMECapabilities ::= SEQUENCE OF SMIMECapability

SMIMECapability ::= SEQUENCE {
        algorithm  ALGORITHM.&id ({SMIMEv3Algorithms}),
        parameters ALGORITHM.&Type ({SMIMEv3Algorithms}{@algorithm})
}

SMIMEv3Algorithms ALGORITHM ::= { ... -- See RFC 2633 -- }
```

[3] This attribute was first mentioned in the document "S/MIME Implementation Guide, Interoperability Profile, Version 1", published on RSA Laboratories' web pages in August, 1995. It was defined as an attribute suitable for use in **SignerInfo** values.

6. Matching rules

This section defines matching rules used in the definition of attributes in this document.

6.1 Case ignore match

The **pkcs9CaseIgnoreMatch** rule compares for equality a presented string with an attribute value of type **PKCS9String**, without regard to the case (upper or lower) of the strings (e.g. “Pkcs” and “PKCS” match).

```
pkcs9CaseIgnoreMatch MATCHING-RULE ::= {
        SYNTAX          PKCS9String {pkcs9-ub-match}
        ID              id-mr-pkcs9CaseIgnoreMatch
}
```

The rule returns **TRUE** if the strings are the same length and corresponding characters are identical except possibly with regard to case.

Where the strings being matched are of different ASN.1 syntax, the comparison proceeds as normal so long as the corresponding characters are in both character sets. Otherwise matching fails.

6.2 Case exact match

The **pkcs9CaseExactMatch** rule compares for equality a presented string with an attribute value of type **PKCS9String**.

```
pkcs9CaseExactMatch MATCHING-RULE ::= {
        SYNTAX          PKCS9String {pkcs-9-ub-match}
        ID              id-mr-pkcs9CaseExactMatch
}
```

The rule is identical to the **pkcs9CaseIgnoreMatch** rule except that case is not ignored.

6.3 Signing time match

The **signingTimeMatch** rule compares for equality a presented value with an attribute value of type **SigningTime**.

```
signingTimeMatch MATCHING-RULE ::= {
        SYNTAX SigningTime
        ID pkcs-9-mr-signingTimeMatch
}
```

The rule returns **TRUE** if the attribute value represents the same time as the presented value. If a time is specified with seconds (or fractional seconds) absent, the number of seconds (fractional seconds) is assumed to be zero

Where the strings being matched are of different ASN.1 syntax, the comparison proceeds as follows:

1. Convert both values to DER-encoded values of type **GeneralizedTime**, coordinated universal time. If this is not possible the matching fails.

2. Compare the strings for equality. The rule returns **TRUE** if and only if the strings are of the same length and corresponding octets are identical.

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-9**.

```
PKCS-9 {iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) modules(0) pkcs-
9(1)}

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

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

IMPORTS

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

ub-name
    FROM UpperBounds upperBounds

OBJECT-CLASS, ATTRIBUTE, MATCHING-RULE, Attribute, top, objectIdentifierMatch
    FROM InformationFramework informationFramework

ALGORITHM, Extensions, Time
    FROM AuthenticationFramework authenticationFramework

DirectoryString, octetStringMatch, caseIgnoreMatch, caseExactMatch,
generalizedTimeMatch, integerMatch, serialNumber
    FROM SelectedAttributeTypes selectedAttributeTypes

SignerInfo
    FROM CryptographicMessageSyntax {iso(1) member-body(2) us(840)
    rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) modules(0) cms(1)}

EncryptedPrivateKeyInfo
    FROM PKCS-8 {iso(1) member-body(2) us(840) rsadsi(113549)
    pkcs(1) pkcs-8(8) modules(1) pkcs-8(1)}

PFX
    FROM PKCS-12 {iso(1) member-body(2) us(840) rsadsi(113549)
    pkcs(1) pkcs-12(12) modules(0) pkcs-12(1)}

PKCS15Token
    FROM PKCS-15 {iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-15(15)
      modules(1) pkcs-15(1)};

-- Upper bounds
```

```
pkcs-9-ub-pkcs9String              INTEGER ::= 255
pkcs-9-ub-emailAddress             INTEGER ::= pkcs-9-ub-pkcs9String
pkcs-9-ub-unstructuredName         INTEGER ::= pkcs-9-ub-pkcs9String
pkcs-9-ub-unstructuredAddress      INTEGER ::= pkcs-9-ub-pkcs9String
pkcs-9-ub-challengePassword        INTEGER ::= pkcs-9-ub-pkcs9String
pkcs-9-ub-friendlyName             INTEGER ::= pkcs-9-ub-pkcs9String
pkcs-9-ub-signingDescription       INTEGER ::= pkcs-9-ub-pkcs9String
pkcs-9-ub-match                    INTEGER ::= pkcs-9-ub-pkcs9String
pkcs-9-ub-pseudonym                INTEGER ::= ub-name
pkcs-9-ub-placeOfBirth             INTEGER ::= ub-name

-- Object Identifiers

pkcs-9 OBJECT IDENTIFIER ::= {iso(1) member-body(2) us(840)
                  rsadsi(113549) pkcs(1) 9}

  -- Main arcs
pkcs-9-mo          OBJECT IDENTIFIER ::= {pkcs-9 0}  -- Modules branch
pkcs-9-oc          OBJECT IDENTIFIER ::= {pkcs-9 24} -- Object class branch
pkcs-9-at          OBJECT IDENTIFIER ::= {pkcs-9 25} -- Attribute branch,
                         -- for new attributes
pkcs-9-sx          OBJECT IDENTIFIER ::= {pkcs-9 26} -- For syntaxes (RFC 2252)
pkcs-9-mr          OBJECT IDENTIFIER ::= {pkcs-9 27} -- Matching rules

  -- Object classes
pkcs-9-oc-pkcsEntity                      OBJECT IDENTIFIER ::= {pkcs-9-oc 1}
pkcs-9-oc-naturalPerson                   OBJECT IDENTIFIER ::= {pkcs-9-oc 2}

  -- Attributes
pkcs-9-at-emailAddress                    OBJECT IDENTIFIER ::= {pkcs-9 1}
pkcs-9-at-unstructuredName                OBJECT IDENTIFIER ::= {pkcs-9 2}
pkcs-9-at-contentType                     OBJECT IDENTIFIER ::= {pkcs-9 3}
pkcs-9-at-messageDigest                   OBJECT IDENTIFIER ::= {pkcs-9 4}
pkcs-9-at-signingTime                     OBJECT IDENTIFIER ::= {pkcs-9 5}
pkcs-9-at-counterSignature                OBJECT IDENTIFIER ::= {pkcs-9 6}
pkcs-9-at-challengePassword               OBJECT IDENTIFIER ::= {pkcs-9 7}
pkcs-9-at-unstructuredAddress             OBJECT IDENTIFIER ::= {pkcs-9 8}
pkcs-9-at-extendedCertificateAttributes   OBJECT IDENTIFIER ::= {pkcs-9 9}

-- Obsolete (?) attribute identifiers, purportedly from "tentative
-- PKCS #9 draft"
-- pkcs-9-at-issuerAndSerialNumber        OBJECT IDENTIFIER ::= {pkcs-9 10}
-- pkcs-9-at-passwordCheck                OBJECT IDENTIFIER ::= {pkcs-9 11}
-- pkcs-9-at-publicKey                    OBJECT IDENTIFIER ::= {pkcs-9 12}

pkcs-9-at-signingDescription              OBJECT IDENTIFIER ::= {pkcs-9 13}
pkcs-9-at-extensionRequest                OBJECT IDENTIFIER ::= {pkcs-9 14}
pkcs-9-at-smimeCapabilities               OBJECT IDENTIFIER ::= {pkcs-9 15}

-- Unused (?)
-- pkcs-9-at-?                            OBJECT IDENTIFIER ::= {pkcs-9 17}
-- pkcs-9-at-?                            OBJECT IDENTIFIER ::= {pkcs-9 18}
-- pkcs-9-at-?                            OBJECT IDENTIFIER ::= {pkcs-9 19}
```

```
pkcs-9-at-friendlyName                  OBJECT IDENTIFIER ::= {pkcs-9 20}
pkcs-9-at-localKeyId                    OBJECT IDENTIFIER ::= {pkcs-9 21}
pkcs-9-at-userPKCS12                    OBJECT IDENTIFIER ::=
                                        {2 16 840 1 113730 3 1 216}
pkcs-9-at-pkcs15Token                   OBJECT IDENTIFIER ::= {pkcs-9-at 1}
pkcs-9-at-encryptedPrivateKeyInfo       OBJECT IDENTIFIER ::= {pkcs-9-at 2}
pkcs-9-at-randomNonce                   OBJECT IDENTIFIER ::= {pkcs-9-at 3}
pkcs-9-at-sequenceNumber                OBJECT IDENTIFIER ::= {pkcs-9-at 4}

  -- IETF PKIX Attribute branch
ietf-at                                 OBJECT IDENTIFIER ::= {1 3 6 1 5 5 7 9}

pkcs-9-at-dateOfBirth                   OBJECT IDENTIFIER ::= {ietf-at 1}
pkcs-9-at-placeOfBirth                  OBJECT IDENTIFIER ::= {ietf-at 2}
pkcs-9-at-gender                        OBJECT IDENTIFIER ::= {ietf-at 3}
pkcs-9-at-countryOfCitizenship          OBJECT IDENTIFIER ::= {ietf-at 4}
pkcs-9-at-countryOfResidence            OBJECT IDENTIFIER ::= {ietf-at 5}

  -- Syntaxes (for use with LDAP accessible directories)
pkcs-9-sx-pkcs9String                   OBJECT IDENTIFIER ::= {pkcs-9-sx 1}
pkcs-9-sx-signingTime                   OBJECT IDENTIFIER ::= {pkcs-9-sx 2}

  -- Matching rules
pkcs-9-mr-caseIgnoreMatch               OBJECT IDENTIFIER ::= {pkcs-9-mr 1}
pkcs-9-mr-caseExactMatch                OBJECT IDENTIFIER ::= {pkcs-9-mr 2}
pkcs-9-mr-signingTimeMatch              OBJECT IDENTIFIER ::= {pkcs-9-mr 3}

  -- Arcs with attributes defined elsewhere
smime                                   OBJECT IDENTIFIER ::= {pkcs-9 16}
  -- Main arc for S/MIME (RFC 2633)
certTypes                               OBJECT IDENTIFIER ::= {pkcs-9 22}
  -- Main arc for certificate types defined in PKCS #12
crlTypes                                OBJECT IDENTIFIER ::= {pkcs-9 23}
  -- Main arc for crl types defined in PKCS #12

  -- Other object identifiers
id-at-pseudonym                         OBJECT IDENTIFIER ::= {id-at 65}

-- Useful types

PKCS9String {INTEGER : maxSize} ::= CHOICE {
      ia5String IA5String (SIZE(1..maxSize)),
      directoryString DirectoryString {maxSize}
}

-- Object classes

pkcsEntity OBJECT-CLASS ::= {
      SUBCLASS OF      { top }
      KIND             auxiliary
      MAY CONTAIN      { PKCS9AttributeSet }
      ID               pkcs-9-oc-pkcsEntity
}
```

```
naturalPerson OBJECT-CLASS ::= {
      SUBCLASS OF     { top }
      KIND            auxiliary
      MAY CONTAIN     { NaturalPersonAttributeSet }
      ID              pkcs-9-oc-naturalPerson
}

-- Attribute sets

PKCS9AttributeSet ATTRIBUTE ::= {
      userPKCS12 |
      pKCS15Token |
      encryptedPrivateKeyInfo,
      ... -- For future extensions
}

NaturalPersonAttributeSet ATTRIBUTE ::= {
      emailAddress |
      unstructuredName |
      unstructuredAddress |
      dateOfBirth |
      placeOfBirth |
      gender |
      countryOfCitizenship |
      countryOfResidence |
      pseudonym |
      serialNumber,
      ... -- For future extensions
}

-- Attributes

userPKCS12 ATTRIBUTE ::= {
      WITH SYNTAX PFX
      ID pkcs-9-at-userPKCS12
}

pKCS15Token ATTRIBUTE ::= {
      WITH SYNTAX PKCS15Token
      ID pkcs-9-at-pkcs15Token
}

encryptedPrivateKeyInfo ATTRIBUTE ::= {
      WITH SYNTAX EncryptedPrivateKeyInfo
      ID pkcs-9-at-encryptedPrivateKeyInfo
}

emailAddress ATTRIBUTE ::= {
      WITH SYNTAX IA5String (SIZE(1..pkcs-9-ub-emailAddress))
      EQUALITY MATCHING RULE pkcs9CaseIgnoreMatch
      ID pkcs-9-at-emailAddress
}
```

```
unstructuredName ATTRIBUTE ::= {
    WITH SYNTAX PKCS9String {pkcs-9-ub-unstructuredName}
    EQUALITY MATCHING RULE pkcs9CaseIgnoreMatch
    ID pkcs-9-at-unstructuredName
}

unstructuredAddress ATTRIBUTE ::= {
    WITH SYNTAX DirectoryString {pkcs-9-ub-unstructuredAddress}
    EQUALITY MATCHING RULE caseIgnoreMatch
    ID pkcs-9-at-unstructuredAddress
}

dateOfBirth ATTRIBUTE ::= {
    WITH SYNTAX GeneralizedTime
    EQUALITY MATCHING RULE generalizedTimeMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-dateOfBirth
}

placeOfBirth ATTRIBUTE ::= {
    WITH SYNTAX DirectoryString {pkcs-9-ub-placeOfBirth}
    EQUALITY MATCHING RULE caseExactMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-placeOfBirth
}

gender ATTRIBUTE ::= {
    WITH SYNTAX PrintableString (SIZE(1) ^ FROM ("M" | "F" | "m" | "f"))
    EQUALITY MATCHING RULE caseIgnoreMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-gender
}

countryOfCitizenship ATTRIBUTE ::= {
    WITH SYNTAX PrintableString (SIZE(2))(CONSTRAINED BY {
    -- Must be a two-letter country acronym in accordance with
    -- ISO/IEC 3166 --})
    EQUALITY MATCHING RULE caseIgnoreMatch
    ID pkcs-9-at-countryOfCitizenship
}

countryOfResidence ATTRIBUTE ::= {
    WITH SYNTAX PrintableString (SIZE(2))(CONSTRAINED BY {
    -- Must be a two-letter country acronym in accordance with
    -- ISO/IEC 3166 --})
    EQUALITY MATCHING RULE caseIgnoreMatch
    ID pkcs-9-at-countryOfResidence
}

pseudonym ATTRIBUTE ::= {
    WITH SYNTAX DirectoryString {pkcs-9-ub-pseudonym}
    EQUALITY MATCHING RULE caseExactMatch
    ID id-at-pseudonym
}
```

```
contentType ATTRIBUTE ::= {
    WITH SYNTAX ContentType
    EQUALITY MATCHING RULE objectIdentifierMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-contentType
}

ContentType ::= OBJECT IDENTIFIER

messageDigest ATTRIBUTE ::= {
    WITH SYNTAX MessageDigest
    EQUALITY MATCHING RULE octetStringMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-messageDigest
}

MessageDigest ::= OCTET STRING

signingTime ATTRIBUTE ::= {
    WITH SYNTAX SigningTime
    EQUALITY MATCHING RULE signingTimeMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-signingTime
}

SigningTime ::= Time -- imported from ISO/IEC 9594-8

randomNonce ATTRIBUTE ::= {
    WITH SYNTAX RandomNonce
    EQUALITY MATCHING RULE octetStringMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-randomNonce
}

RandomNonce ::= OCTET STRING (SIZE(4..MAX)) -- At least four bytes long

sequenceNumber ATTRIBUTE ::= {
    WITH SYNTAX SequenceNumber
    EQUALITY MATCHING RULE integerMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-sequenceNumber
}

SequenceNumber ::= INTEGER (1..MAX)

counterSignature ATTRIBUTE ::= {
    WITH SYNTAX SignerInfo
    ID pkcs-9-at-counterSignature
}

challengePassword ATTRIBUTE ::= {
    WITH SYNTAX PKCS9String {pkcs-9-ub-challengePassword}
    EQUALITY MATCHING RULE pkcs9CaseExactMatch
```

```
    SINGLE VALUE TRUE
    ID pkcs-9-at-challengePassword
}

extensionRequest ATTRIBUTE ::= {
    WITH SYNTAX ExtensionRequest
    SINGLE VALUE TRUE
    ID pkcs-9-at-extensionRequest
}

ExtensionRequest ::= Extensions

extendedCertificateAttributes ATTRIBUTE ::= {
    WITH SYNTAX SET OF Attribute
    SINGLE VALUE TRUE
    ID pkcs-9-at-extendedCertificateAttributes
}

friendlyName ATTRIBUTE ::= {
    WITH SYNTAX BMPString (SIZE(1..pkcs-9-ub-friendlyName))
    EQUALITY MATCHING RULE caseIgnoreMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-friendlyName
}

localKeyId ATTRIBUTE ::= {
    WITH SYNTAX OCTET STRING
    EQUALITY MATCHING RULE octetStringMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-localKeyId
}

signingDescription ATTRIBUTE ::= {
    WITH SYNTAX DirectoryString {pkcs-9-ub-signingDescription}
    EQUALITY MATCHING RULE caseIgnoreMatch
    SINGLE VALUE TRUE
    ID pkcs-9-at-signingDescription
}

smimeCapabilities ATTRIBUTE ::= {
    WITH SYNTAX SMIMECapabilities
    SINGLE VALUE TRUE
    ID pkcs-9-at-smimeCapabilities
}

SMIMECapabilities ::= SEQUENCE OF SMIMECapability

SMIMECapability ::= SEQUENCE {
    algorithm  ALGORITHM.&id ({SMIMEv3Algorithms}),
    parameters ALGORITHM.&Type ({SMIMEv3Algorithms}{@algorithm})
}

SMIMEv3Algorithms ALGORITHM ::= {...-- See RFC 2633 --}
```

```
-- Matching rules

pkcs9CaseIgnoreMatch MATCHING-RULE ::= {
      SYNTAX PKCS9String {pkcs-9-ub-match}
      ID pkcs-9-mr-caseIgnoreMatch
}

pkcs9CaseExactMatch MATCHING-RULE ::= {
      SYNTAX PKCS9String {pkcs-9-ub-match}
      ID pkcs-9-mr-caseExactMatch
}

signingTimeMatch MATCHING-RULE ::= {
      SYNTAX SigningTime
      ID pkcs-9-mr-signingTimeMatch
}

END
```

B. BNF schema summary

This appendix provides augmented BNF [2] definitions of the object class and most attribute types specified in this document along with their associated syntaxes and matching rules. The ABNF definitions have been done in accordance with [21], in an attempt to ease integration with LDAP-accessible Directory systems. Lines have been folded in some cases to improve readability.

B.1 Syntaxes

This section defines all syntaxes that are used in this document.

B.1.1 PKCS9String

```
(
      1.2.840.113549.1.9.26.1
      DESC 'PKCS9String'
)
```

The encoding of a value in this syntax is the string value itself.

B.1.2 SigningTime

```
(
      1.2.840.113549.1.9.26.2
      DESC 'SigningTime'
)
```

Values in this syntax are encoded as printable strings, represented as specified in [5]. Note that the time zone must be specified. For example, "`199412161032Z`"

B.2 Object Classes

B.2.1 pkcsEntity

```
(
      1.2.840.113549.1.9.24.1
      NAME 'pkcsEntity'
      SUP top
      AUXILIARY
      MAY (
      userPKCS12 $ pKCS15Token $ encryptedPrivateKeyInfo
      )
)
```

B.2.2 naturalPerson

```
(
      1.2.840.113549.1.9.24.2
      NAME 'naturalPerson'
      SUP top
      AUXILIARY
      MAY (
      emailAddress $ unstructuredName $ unstructuredAddress $
      dateOfBirth & placeOfBirth & gender & countryOfCitizenship &
      countryOfResidence & pseudonym & serialNumber
      )
)
```

B.3 Attribute types

B.3.1 userPKCS12

This attribute is to be stored and requested in binary form, as `userPKCS12;binary`. The attribute values are **PFX** PDUs stored as binary (BER- or DER-encoded) data.

```
(
      2.16.840.1.113730.3.1.216
      NAME 'userPKCS12'
      DESC 'PKCS #12 PFX PDU for exchange of personal information'
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
)
```

B.3.2 pKCS15Token

This attribute is to be stored and requested in binary form, as `pKCS15Token;binary`. The attribute values are **PKCS15Token** PDUs stored as binary (BER- or DER-encoded) data.

```
(
      1.2.840.113549.1.9.25.1
      NAME 'pKCS15Token'
      DESC 'PKCS #15 token PDU'
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
)
```

B.3.3 encryptedPrivateKeyInfo

This attribute is to be stored and requested in binary form, as `encryptedPrivateKeyInfo;binary`. The attribute values are **EncryptedPrivateKeyInfo** PDUs stored as binary (BER- or DER-encoded) data.

```
(
      1.2.840.113549.1.9.25.2
      NAME 'encryptedPrivateKeyInfo'
      DESC 'PKCS #8 encrypted private key info'
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
)
```

B.3.4 emailAddress

```
(
      1.2.840.113549.1.9.1
      NAME 'emailAddress'
      DESC 'Email address'
      EQUALITY pkcs9CaseIgnoreMatch
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
)
```

B.3.5 unstructuredName

```
(
      1.2.840.113549.1.9.2
      NAME 'unstructuredName'
      DESC 'PKCS #9 unstructured name'
      EQUALITY pkcs9CaseIgnoreMatch
      SYNTAX 1.2.840.113549.1.9.26.1
)
```

B.3.6 unstructuredAddress

```
(
      1.2.840.113549.1.9.8
      NAME 'unstructuredAddress'
      DESC 'PKCS #9 unstructured address'
      EQUALITY caseIgnoreMatch
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
)
```

B.3.7 dateOfBirth

```
(
      1.3.6.1.5.5.7.9.1
      NAME 'dateOfBirth'
      DESC 'Date of birth'
      EQUALITY generalizedTimeMatch
```

```
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
      SINGLE-VALUE
)
```

B.3.8 placeOfBirth

```
(
      1.3.6.1.5.5.7.9.2
      NAME 'placeOfBirth'
      DESC 'Place of birth'
      EQUALITY caseExactMatch
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
      SINGLE-VALUE
)
```

B.3.9 gender

```
(
      1.3.6.1.5.5.7.9.3
      NAME 'gender'
      DESC 'Gender'
      EQUALITY caseIgnoreMatch
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
      SINGLE-VALUE
)
```

B.3.10 countryOfCitizenship

```
(
      1.3.6.1.5.5.7.9.4
      NAME 'countryOfCitizenship'
      DESC 'Country of citizenship'
      EQUALITY caseIgnoreMatch
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
)
```

B.3.11 countryOfResidence

```
(
      1.3.6.1.5.5.7.9.5
      NAME 'countryOfResidence'
      DESC 'Country of residence'
      EQUALITY caseIgnoreMatch
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
)
```

B.3.12 pseudonym

```
(
      2.5.4.65
      NAME 'pseudonym'
      DESC 'Pseudonym'
      EQUALITY caseExactMatch
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
)
```

B.3.13 contentType

In the (highly unlikely) event of this attribute being stored in a Directory it is to be stored and requested in binary form, as `contentType;binary`. Attribute values shall be **OCTET STRING**s stored as binary (BER- or DER-encoded) data.

```
(
      1.2.840.113549.1.9.3
      NAME 'contentType'
      DESC 'PKCS #7 content type attribute'
      EQUALITY objectIdentifierMatch
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.38
      SINGLE-VALUE
)
```

B.3.14 messageDigest

In the (highly unlikely) event of this attribute being stored in a Directory it is to be stored and requested in binary form, as `messageDigest;binary`. Attribute values shall be **OCTET STRING**s stored as binary (BER- or DER-encoded) data.

```
(
      1.2.840.113549.1.9.4
      NAME 'messageDigest'
      DESC 'PKCS #7 mesage digest attribute'
      EQUALITY octetStringMatch
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
      SINGLE-VALUE
)
```

B.3.15 signingTime

```
(
      1.2.840.113549.1.9.5
      NAME 'signingTime'
      DESC 'PKCS #7 signing time'
      EQUALITY signingTimeMatch
      SYNTAX 1.2.840.113549.1.9.26.2
      SINGLE-VALUE
)
```

B.3.16 counterSignature

In the (highly unlikely) event that this attribute is to be stored in a directory, it is to be stored and requested in binary form, as `counterSignature;binary`. Attribute values shall be stored as binary (BER- or DER-encoded) data.

```
(
      1.2.840.113549.1.9.6
      NAME 'counterSignature'
      DESC 'PKCS #7 countersignature'
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
)
```

B.3.17 challengePassword

```
(
      1.2.840.113549.1.9.7
      NAME 'challengePassword'
      DESC 'Challenge password for certificate revocations'
      EQUALITY pkcs9CaseExactMatch
      SYNTAX 1.2.840.113549.1.9.26.1
      SINGLE-VALUE
)
```

Note – It is not recommended to store unprotected values of this attribute in a directory.

B.4 Matching rules

B.4.1 pkcs9CaseIgnoreMatch

```
(
      1.2.840.113549.1.9.27.1
      NAME 'pkcs9CaseIgnoreMatch'
      SYNTAX 1.2.840.113549.1.9.26.1
)
```

B.4.2 pkcs9CaseExactMatch

```
(
      1.2.840.113549.1.9.27.2
      NAME 'pkcs9CaseExactMatch'
      SYNTAX 1.2.840.113549.1.9.26.1
)
```

B.4.3 signingTimeMatch

```
(
      1.2.840.113549.1.9.27.3
      NAME 'signingTimeMatch'
      SYNTAX 1.2.840.113549.1.9.26.2
)
```

C. 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.

D. Revision history

Version 1.0

Version 1.0 was part of the June 3, 1991 initial public release of PKCS. Version 1.0 was also published as NIST/OSI Implementors' Workshop document SEC-SIG-91-24.

Version 1.1

Version 1.1 incorporated several editorial changes, including updates to the references and the addition of a revision history. The following substantive changes were made:

- Section 6: **challengePassword**, **unstructuredAddress**, and **extendedCertificateAttributes** attribute types were added
- Section 7: **challengePassword**, **unstructuredAddress**, and **extendedCertificateAttributes** object identifiers were added

Version 2.0

Version 2.0 incorporates several editorial changes as well. In addition, the following substantive changes has been made:

- Addition of a Section defining two new auxiliary object classes, **pkcsEntity** and **naturalPerson**
- Addition of several new attribute types for use in conjunction with these object classes and elsewhere
- Update of all ASN.1 to be in line with the 1997 version of this syntax
- Addition a "compilable" ASN.1 module
- Addition, in accordance with [21], an ABNF description of all attributes and object classes
- Addition of an intellectual property considerations section

E. References

[1] S. Bradner. *RFC 2119: Key words for use in RFCs to Indicate Requirement Levels*. IETF, March 1997.

[2] D. Crocker, P. Overell. *RFC 2234: Augmented BNF for Syntax Specifications: ABNF*. IETF, November 1997.

[3] R. Housley. *RFC 2630: Cryptographic Message Syntax CMS*. IETF, June 1999.

[4] *ISO/IEC 3166-1:Codes for the representation of names of countries and their subdivisions – Part 1: Country codes*. 1997.

[5] *ISO/IEC 8824-1:1999: Information technology — Abstract Syntax Notation One (ASN.1) — Specification of basic notation.*1999.

[6] *ISO/IEC 8825-1:1999: Information technology – ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*. 1999.

[7] *ISO/IEC 9594-2:1997: Information technology – Open Systems Interconnection – The Directory: Models*. 1997.

[8] *ISO/IEC 9594*-6:1997: *Information technology – Open Systems Interconnection – The Directory: Selected attribute types*. 1997.

[9] *ISO/IEC 9594-7:1997: Information technology – Open Systems Interconnection – The Directory: Selected object classes*. 1997.

[10] *ISO/IEC 9594-8:1997: Information technology – Open Systems Interconnection – The Directory: Authentication framework*. 1997.

[11] *ISO/IEC 10646-1: Information Technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane*. 1993.

[12] B. Ramsdell. *RFC 2633*: *S/MIME Version 3 Message Specification*. IETF, June 1999.

[13] RSA Laboratories. *PKCS #6: Extended-Certificate Syntax Standard*. Version 1.5, November 1993.

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

[15] RSA Laboratories. *PKCS #8: Private-Key Information Syntax Standard*. Version 1.2, November 1993.

[16] RSA Laboratories. *PKCS #10: Certification Request Syntax Standard*. Version 1.0, November 1993.

[17] RSA Laboratories. *PKCS #12: Personal Information Exchange Syntax Standard*. Version 1.0, June 1999.

[18] RSA Laboratories. *PKCS #15: Cryptographic Token Information Format Standard*. Version 1.1 (Draft), December 1999.

[19] S. Santesson, W. Polk, P. Barzin, M. Nystrom. *Internet X.509 Public Key Infrastructure – Qualified Certificates Profile*. IETF work in progress, February 2000.

[20] M. Smith. *Definition of the inetOrgPerson LDAP Object Class*. IETF work in progress, January 2000.

[21] M. Wahl, A. Coulbeck, T. Howes, S. Kille. *RFC 2252: Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions*. IETF, December 1997.

[22] M. Wahl, T. Howes, S. Kille. *RFC 2251: Lightweight Directory Access Protocol (v3)*. IETF, December 1997.

F. 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