

KEYPROV Working Group
Internet Draft
Intended Status: Standards Track
Expires: February 9, 2010

Sean Turner
IECA
Russ Housley
Vigil Security
August 9, 2010

Symmetric Key Package Content Type
draft-ietf-keyprov-symmetrickeyformat-11.txt

Abstract

This document defines the symmetric key format content type. It is transport independent. The Cryptographic Message Syntax can be used to digitally sign, digest, authenticate, or encrypt this content type.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/1id-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>

This Internet-Draft will expire on February 9, 2010.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction.....	3
1.1.	Requirements Terminology.....	3
1.2.	ASN.1 Syntax Notation.....	3
2.	Symmetric Key Package Content Type.....	3
3.	PSKC Attributes.....	5
3.1.	PSKC Key Package Attributes.....	5
3.1.1.	Device Information Attributes.....	5
3.1.2.	Cryptographic Module Information Attributes.....	8
3.2.	PSKC Key Attributes.....	8
3.2.1.	Key Identifier.....	9
3.2.2.	Algorithm.....	9
3.2.3.	Issuer.....	9
3.2.4.	Key Profile Identifier.....	9
3.2.5.	Key Reference Identifier.....	10
3.2.6.	Friendly Name.....	10
3.2.7.	Algorithm Parameters.....	10
3.2.8.	Counter.....	13
3.2.9.	Time.....	13
3.2.10.	Time Interval.....	13
3.2.11.	Time Drift.....	13
3.2.12.	Value MAC.....	14
3.2.13.	Key User Id.....	14
3.3.	Key Policy Attributes.....	15
3.3.1.	Key Start Date.....	15
3.3.2.	Key Expiry Date.....	15
3.3.3.	Number of Transactions.....	15
3.3.4.	Key Usage.....	16
3.3.5.	PIN Policy.....	17
4.	Key Encoding.....	18
4.1.	AES Key Encoding.....	18
4.2.	Triple DES Key Encoding.....	18

5.	Security Considerations.....	19
6.	IANA Considerations.....	19
7.	References.....	20
7.1.	Normative References.....	20
7.2.	Informative References.....	22
APPENDIX A: ASN.1 Module.....		22
A.1.	Symmetric Key Package ASN.1 Module.....	22
A.2.	PSKC ASN.1 Module.....	24

[1.](#) Introduction

This document defines the symmetric key format content type. It is transport independent. The Cryptographic Message Syntax (CMS) [[RFC5652](#)] can be used to digitally sign, digest, authenticate, or encrypt this content type.

The use cases that motivated the attributes in this work are elaborated in [[PSKC](#)]. They are omitted to avoid duplication.

This document also includes Abstract Syntax Notation One (ASN.1) definitions of the Extensible Markup Language (XML) element and attributes defined in [[PSKC](#)].

[1.1.](#) Requirements Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119](#)].

[1.2.](#) ASN.1 Syntax Notation

The key package is defined using the ASN.1 [[X.680](#)], [[X.681](#)], [[X.682](#)], and [[X.683](#)].

[2.](#) Symmetric Key Package Content Type

The symmetric key package content type is used to transfer one or more plaintext symmetric keys from one party to another. A symmetric key package MAY be encapsulated in one or more CMS protecting content types. This content type MUST be Distinguished Encoding Rules (DER) encoded [[X.690](#)].

The symmetric key package content type has the following syntax:

```
ct-symmetric-key-package CONTENT-TYPE ::=
    { SymmetricKeyPackage IDENTIFIED BY id-ct-KP-sKeyPackage }
```



```
id-ct-KP-sKeyPackage OBJECT IDENTIFIER ::=
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
    smime(16) ct(1) 25 }

SymmetricKeyPackage ::= SEQUENCE {
  version          KeyPkgVersion DEFAULT v1,
  sKeyPkgAttrs    [0] SEQUENCE SIZE (1..MAX) OF Attribute
                                     {{ SKeyPkgAttributes }} OPTIONAL,
  sKeys           SymmetricKeys,
  ... }

SymmetricKeys ::= SEQUENCE SIZE (1..MAX) OF OneSymmetricKey

OneSymmetricKey ::= SEQUENCE {
  sKeyAttrs       SEQUENCE SIZE (1..MAX) OF Attribute
                                     {{ SKeyAttributes }} OPTIONAL,
  sKey            OCTET STRING OPTIONAL }
  ( WITH COMPONENTS { ..., sKeyAttrs PRESENT } |
    WITH COMPONENTS { ..., sKey PRESENT } )

KeyPkgVersion ::= INTEGER { v1(1) } ( v1, ... )
```

The SymmetricKeyPackage fields are used as follows:

- version identifies version of the symmetric key package content structure. For this version of the specification, the default value, v1, MUST be used.
- sKeyPkgAttrs optionally provides attributes that apply to all of the symmetric keys in the package. The SKeyPkgAttributes information object set restricts the attributes allowed in sKeyPkgAttrs. If an attribute appears here, then it MUST NOT also be included in sKeyAttrs.
- sKeys contains a sequence of OneSymmetricKey values. This structure is discussed below.

The OneSymmetricKey fields are used as follows:

- sKeyAttrs optionally provides attributes that apply to one symmetric key. The SKeyAttributes information object set restricts the attributes permitted in sKeyAttrs. If an attribute appears here, then it MUST NOT also be included in sKeyPkgAttrs.
- sKey optionally contains the key value encoded as an OCTET STRING.

The OneSymmetricKey field MUST include either sKeyAttrs, or sKey, or sKeyAttrs and sKey.

3. PSKC Attributes

The following attributes are defined to assist those using the symmetric key package defined in this document as part of a Dynamic Symmetric Key Provision Protocol (DSKPP) [DSKPP] with Portable Symmetric Key Container (PSKC) attributes. [PSKC] should be consulted for the definitive attribute descriptions. The attributes fall in to three categories. The first category includes attributes that apply to a key package, and these attributes will generally appear in sKeyPkgAttrs. The second category includes attributes that apply to a particular key, and these attributes will generally appear in sKeyAttrs. The third category includes attributes that apply to a key policy. Of the attributes defined, only the Key Identifier (Section 3.2.1) and Algorithm (Section 3.2.2) key attributes MUST be included. All other attributes are OPTIONAL.

Like PSKC, the Symmetric Key Content Type supports extensibility. Primarily this is accomplished through the definition and inclusion of new attributes, but in some instances where the attribute contains more than one type the ASN.1 "... " extensibility mechanism is employed.

A straightforward approach to conversion from XML types to ASN.1 is employed. The <xs:string> type converts to UTF8String; the XML <xs:dateTime> type converts to GeneralizedTime; and the XML integer types convert to INTEGER or BinaryTime [RFC4049].

3.1. PSKC Key Package Attributes

PSKC key package attributes apply to an entire key package. These attributes can be categorized by two different attribute collections: device information and cryptographic module attributes. All of these key package attributes are OPTIONAL.

3.1.1. Device Information Attributes

Device Information attributes when taken together MUST uniquely identify a device to which the Symmetric Key Package is provisioned.

3.1.1.1. Manufacturer

The Manufacturer attribute indicates the manufacturer of the device. Values for Manufacturer MUST be taken from either [OATHMAN] prefixes (i.e., the left column) or from IANA Private Enterprise Number

Registry [[IANAPENREG](#)], using the Organisation value. When the value is taken from [[OATHMAN](#)] "oath." MUST be prepended to the value (e.g., "oath.<values from [[OATHMAN](#)]>"). When the value is taken from [[IANAPENREG](#)] "iana." MUST be prepended to the value (e.g., "iana.<Organisation value from [[IANAPENREG](#)]>"). The attribute definition is as follows:

```
at-pskc-manufacturer ATTRIBUTE ::= {  
    TYPE UTF8String IDENTIFIED BY id-pskc-manufacturer }  
  
id-pskc-manufacturer OBJECT IDENTIFIER ::= { id-pskc 1 }
```

3.1.1.2. Serial Number

The Serial Number attribute indicates the serial number of the device. The attribute definition is as follows:

```
at-pskc-serialNo ATTRIBUTE ::= {  
    TYPE UTF8String IDENTIFIED BY id-pskc-serialNo }  
  
id-pskc-serialNo OBJECT IDENTIFIER ::= { id-pskc 2 }
```

3.1.1.3. Model

The Model attribute indicates the model of the device. The attribute definition is as follows:

```
at-pskc-model ATTRIBUTE ::= {  
    TYPE UTF8String IDENTIFIED BY id-pskc-model }  
  
id-pskc-model OBJECT IDENTIFIER ::= { id-pskc 3 }
```

3.1.1.4. Issue Number

The Issue Number attribute contains an issue number to distinguish between two devices with the same serial number. The attribute definition is as follows:

```
at-pskc-issueNo ATTRIBUTE ::= {  
    TYPE UTF8String IDENTIFIED BY id-pskc-issueNo }  
  
id-pskc-issueNo OBJECT IDENTIFIER ::= { id-pskc 4 }
```

3.1.1.5. Device Binding

The Device Binding attribute provides an opaque identifier that allows keys to be bound to the device or to a class of devices.

When loading keys into a device, the attribute's value MUST be checked against information provided to the user via out-of-band mechanisms. The implementation then ensures that the correct device or class of device is being used with respect to the provisioned key.

```
at-pskc-deviceBinding ATTRIBUTE ::= {  
    TYPE UTF8String IDENTIFIED BY id-pskc-deviceBinding }  
  
id-pskc-deviceBinding OBJECT IDENTIFIER ::= { id-pskc 5 }
```

3.1.1.6. Device Start Date

When included in sKeyPkgAttrs, the Device Start Date attribute indicates the start date for a device. The date MUST be represented in a form that matches the dateTime production in "canonical representation" [[XMLSCHEMA](#)]. Implementations SHOULD NOT rely on time resolution finer than milliseconds and MUST NOT generate time instants that specify leap seconds. Keys that are on the device SHOULD only be used when the current date is on or after the device start date. The attribute definition is as follows:

```
at-pskc-deviceStartDate ATTRIBUTE ::= {  
    TYPE GeneralizedTime IDENTIFIED BY id-pskc-deviceStartDate }  
  
id-pskc-deviceStartDate OBJECT IDENTIFIER ::= { id-pskc 6 }
```

Note that usage enforcement of the keys with respect to the dates MAY only happen on the validation server as some devices such as smart cards do not have an internal clock. Systems thus SHOULD NOT rely upon the device to enforce key usage date restrictions.

3.1.1.7. Device Expiry Date

When included in sKeyPkgAttrs, the Device Expiry Date attribute indicates the expiry date for a device. The date MUST be represented in a form that matches the dateTime production in "canonical representation" [[XMLSCHEMA](#)]. Implementations SHOULD NOT rely on time resolution finer than milliseconds and MUST NOT generate time instants that specify leap seconds. Keys that are on the device SHOULD only be used when the current date is before the device expiry date. The attribute definition is as follows:

```
at-pskc-deviceExpiryDate ATTRIBUTE ::= {  
    TYPE GeneralizedTime IDENTIFIED BY id-pskc-deviceExpiryDate }  
  
id-pskc-deviceExpiryDate OBJECT IDENTIFIER ::= { id-pskc 7 }
```


Note that usage enforcement of the keys with respect to the dates MAY only happen on the validation server as some devices such as smart cards do not have an internal clock. Systems thus SHOULD NOT rely upon the device to enforce key usage date restrictions.

3.1.1.8. Device User Id

The Device User Id attribute indicates the user with whom the device is associated with using a distinguished name, as defined in [\[RFC4514\]](#). For example: UID=jsmith,DC=example,DC=net. The attribute definition is as follows:

```
at-pskc-deviceUserId ATTRIBUTE ::= {  
  
    TYPE UTF8String IDENTIFIED BY id-pskc-deviceUserId }  
  
id-pskc-deviceUserId OBJECT IDENTIFIER ::= { id-pskc 26 }
```

As specified in [\[PSKC\]](#), there are no semantics associated with this element, i.e., there are no checks enforcing that only a specific user can use this device. As such, this element is for informational purposes only.

3.1.2. Cryptographic Module Information Attributes

Cryptographic Module attributes uniquely identify a cryptographic module. This is useful when the device contains more than one cryptographic module. At this time only one attribute is defined.

3.1.2.1. Cryptographic Module Identifier

When included in sKeyPkgAttrs, the Cryptographic Module Identifier attribute uniquely identifies the cryptographic module to which the key is being or was provisioned. The attribute definition is as follows:

```
at-pskc-moduleId ATTRIBUTE ::= {  
    TYPE UTF8String IDENTIFIED BY id-pskc-moduleId }  
  
id-pskc-moduleId OBJECT IDENTIFIER ::= { id-pskc 8 }
```

3.2. PSKC Key Attributes

PSKC key attributes apply to a specific key. As noted earlier, the Key Identifier (Sec 3.2.1) and Algorithm (Sec 3.2.2) key attributes are REQUIRED. All other attributes are OPTIONAL.

3.2.1. Key Identifier

When included in sKeyAttrs, the Key Identifier attribute identifies the key in the context of key provisioning exchanges between two parties. This means that if PSKC is used in multiple interactions between a sending and receiving party, using different containers referencing the same keys, the KeyId MUST use the same KeyId values (e.g. after initial provisioning, if a system wants to update key meta data values in the other system the KeyId value of the key where the meta data is to be updates MUST be the same as the original KeyId value provisioned). The attribute definition is as follows:

```
at-pskc-keyId ATTRIBUTE ::= {  
  TYPE UTF8String IDENTIFIED BY id-pskc-keyId }  
  
id-pskc-keyId OBJECT IDENTIFIER ::= { id-pskc 9 }
```

3.2.2. Algorithm

The Algorithm attribute uniquely identifies the PSKC algorithm profile. [\[PSKC\]](#) defines two algorithm profiles "HOTP" and "PIN". The attribute definition is as follows:

```
at-pskc-algorithm ATTRIBUTE ::= {  
  TYPE UTF8String IDENTIFIED BY id-pskc-algorithm }  
  
id-pskc-algorithm OBJECT IDENTIFIER ::= { id-pskc 10 }
```

3.2.3. Issuer

The Issuer attribute names the entity that issued the key. The attribute definition is as follows:

```
at-pskc-issuer ATTRIBUTE ::= {  
  TYPE UTF8String IDENTIFIED BY id-pskc-issuer }  
  
id-pskc-issuer OBJECT IDENTIFIER ::= { id-pskc 11 }
```

3.2.4. Key Profile Identifier

The Key Profile Identifier attribute carries a unique identifier used between the sending and receiving parties to establish a set of key attribute values that are not transmitted within the container but agreed between the two parties out of band. This attribute will then represent the unique reference to a set of key attribute values.


```
at-pskc-keyProfileId ATTRIBUTE ::= {  
    TYPE UTF8String IDENTIFIED BY id-pskc-keyProfileId }  
  
id-pskc-keyProfileId OBJECT IDENTIFIER ::= { id-pskc 12 }
```

3.2.5. Key Reference Identifier

The Key Reference attribute refers to an external key to be used with a key derivation scheme and no specific key value (secret) is transported but only the reference to the external master key is used (e.g., the PKCS#11 key label).

```
at-pskc-keyReference ATTRIBUTE ::= {  
    TYPE UTF8String IDENTIFIED BY id-pskc-keyReference }  
  
id-pskc-keyReference OBJECT IDENTIFIER ::= { id-pskc 13 }
```

3.2.6. Friendly Name

The Friendly Name attribute contains a human readable name for the secret key. The attribute definition is as follows:

```
at-pskc-friendlyName ATTRIBUTE ::= {  
    TYPE FriendlyName IDENTIFIED BY id-pskc-friendlyName }  
  
id-pskc-friendlyName OBJECT IDENTIFIER ::= { id-pskc 14 }
```

The Friendly Name attribute has the following syntax:

```
FriendlyName ::= SEQUENCE {  
    friendlyName UTF8String,  
    friendlyNameLangTag UTF8String OPTIONAL }
```

The text is encoded in UTF-8 [[RFC3629](#)], which accommodates most of the world's writing systems. The friendlyNameLangTag field identifies the language used to express the friendlyName. When friendlyNameLangTag is absent, English, whose associated language tag is "en", is used. The value of the friendlyNameLangTag MUST be a language tag as described in [[RFC5646](#)].

3.2.7. Algorithm Parameters

The Algorithm Parameters attribute contains parameters that influence the result of the algorithmic computation, for example response truncation and format in One-Time Password (OTP) and Challenge-Response (CR) algorithms.


```
at-pskc-algorithmParameters ATTRIBUTE ::= {  
  TYPE PSKCAAlgorithmParameters  
  IDENTIFIED BY id-pskc-algorithmParams }
```

```
id-pskc-algorithmParams OBJECT IDENTIFIER ::= { id-pskc 15 }
```

The Algorithm Parameters attribute has the following syntax:

```
PSKCAAlgorithmParameters ::= CHOICE {  
  suite                UTF8String OPTIONAL,  
  challengeFormat      [0] ChallengeFormat,  
  responseFormat       [1] ResponseFormat,  
  ... }
```

```
ChallengeFormat ::= SEQUENCE {  
  encoding      Encoding,  
  checkDigit    BOOLEAN DEFAULT FALSE,  
  min           INTEGER (0..MAX),  
  max           INTEGER (0..MAX),  
  ... }
```

```
Encoding ::= UTF8STRING ("DECIMAL" | "HEXADECIMAL" |  
  "ALPHANUMERIC" | "BASE64" | "BINARY")
```

```
ResponseFormat ::= SEQUENCE {  
  encoding      Encoding,  
  length        INTEGER (0..MAX),  
  checkDigit    BOOLEAN DEFAULT FALSE,  
  ... }
```

The fields in PSKCAAlgorithmParameters have the following meanings:

- o Suite defines additional characteristics of the algorithm used, which are algorithm specific. For example in HMAC based OTP algorithm it could designate the strength of the hash algorithm used (SHA1, SHA256, etc). Please refer to algorithm profile specification [[PSKC](#)] for the exact semantic of the value for each algorithm profile.
- o ChallengeFormat defines the characteristics of the challenge in a CR usage scenario whereby the following fields are defined:
 - o encoding specifies the encoding of the challenge accepted by the device and MUST be one of the following values: DECIMAL, HEXADECIMAL, ALPHANUMERIC, BASE64, or BINARY. The BASE64 encoding is done as in [Section 4 of \[RFC4648\]](#).

- o `checkDigit` indicates whether a device needs to check the appended Luhn check digit, as defined in [[LUHN](#)], contained in a challenge. The `checkDigit` MUST NOT be present if the encoding value is anything other than 'DECIMAL'. A value of TRUE indicates that the device will check the appended Luhn check digit in a provided challenge. A value of FALSE indicates that the device will not check the appended Luhn check digit in the challenge.
- o `min` defines the minimum size of the challenge accepted by the device for CR mode. If encoding is 'DECIMAL', 'HEXADECIMAL' or 'ALPHANUMERIC' this value indicates the minimum number of digits/characters. If encoding is 'BASE64' or 'BINARY', this value indicates the minimum number of bytes of the unencoded value.
- o `max` defines the maximum size of the challenge accepted by the device for CR mode. If encoding is 'DECIMAL', 'HEXADECIMAL' or 'ALPHANUMERIC' this value indicates the maximum number of digits/characters. If the encoding is 'BASE64' or 'BINARY', this value indicates the maximum number of bytes of the unencoded value.
- o `ResponseFormat` defines the characteristics of the result of a computation and defines the format of the OTP or the response to a challenge. For cases where the key is a personal identification number (PIN) value, this element contains the format of the PIN itself (e.g., DECIMAL, length 4 for a 4 digit PIN). The following fields are defined:
 - o `encoding` specifies the encoding of the response generated by the device and MUST be one of the following values: DECIMAL, HEXADECIMAL, ALPHANUMERIC, BASE64, or BINARY. BASE64 is defined as in [Section 4 of \[RFC4648\]](#).
 - o `length` defines the length of the response generated by the device. If encoding is 'DECIMAL', 'HEXADECIMAL' or 'ALPHANUMERIC' this value indicates the number of digits/characters. If encoding is 'BASE64' or 'BINARY', this value indicates the number of bytes of the unencoded value.
 - o `checkDigit` indicates whether the device needs to append a Luhn check digit, as defined in [[LUHN](#)], to the response. This is only valid if encoding attribute is 'DECIMAL'. If the value is TRUE then the device will append a Luhn check digit to the response. If the value is FALSE, then the device will not append a Luhn check digit to the response.

3.2.8. Counter

The Counter attribute contains the event counter for event-based OTP algorithms. The attribute definition is as follows:

```
at-pskc-counter ATTRIBUTE ::= {  
    TYPE INTEGER(0..MAX) IDENTIFIED BY id-pskc-counter }  
  
id-pskc-counter OBJECT IDENTIFIER ::= { id-pskc 16 }
```

3.2.9. Time

The Time attribute conveys the time for time-based OTP algorithms. If the Time Interval attribute is included, then this element carries the number of time intervals passed for a specific start point. If time interval is used, then this element carries the number of time intervals passed from a specific start point, normally algorithm dependent. It uses the BinaryTime syntax from [[RFC4049](#)]. The attribute definition is as follows:

```
at-pskc-time ATTRIBUTE ::= {  
    TYPE BinaryTime IDENTIFIED BY id-pskc-time }  
  
id-pskc-time OBJECT IDENTIFIER ::= { id-pskc 17 }
```

3.2.10. Time Interval

The Time Interval attribute conveys the time interval value for time-based OTP algorithms in seconds (e.g., a value of 30 for this would indicate a time interval of 30 seconds). It is an integer. The attribute definition is as follows:

```
at-pskc-timeInterval ATTRIBUTE ::= {  
    TYPE INTEGER (0..MAX) IDENTIFIED BY id-pskc-timeInterval }  
  
id-pskc-timeInterval OBJECT IDENTIFIER ::= { id-pskc 18 }
```

3.2.11. Time Drift

The Time Drift attribute contains the device clock drift value for time-based OTP algorithms. It is an integer either positive or negative that indicates the number of time intervals that a validation server has established the device clock drifted after the last successful authentication. The attribute definition is as follows:


```
at-pskc-timeDrift ATTRIBUTE ::= {  
    TYPE INTEGER (0..MAX) IDENTIFIED BY id-pskc-timeDrift }  
  
id-pskc-timeDrift OBJECT IDENTIFIER ::= { id-pskc 19 }
```

3.2.12. Value MAC

The Value MAC attribute is a Message Authentication Code (MAC) generated from the encrypted value in case the encryption algorithm does not support integrity checks (e.g., AES-CBC does not provide integrity while AES Key Wrap with MLI does). The attribute definition is as follows:

```
at-pskc-valueMAC ATTRIBUTE ::= {  
    TYPE ValueMac IDENTIFIED BY id-pskc-valueMAC }  
  
id-pskc-valueMAC OBJECT IDENTIFIER ::= { id-pskc 20 }  
  
ValueMac ::= SEQUENCE {  
    macAlgorithm UTF8String,  
    mac          UTF8String }
```

The fields in ValueMac have the following meanings:

- o macAlgorithm identifies the MAC algorithm used to generate the value placed in digest.
- o mac is the base64 encoded, as specified in [Section 4 of \[RFC4648\]](#), mac value.

3.2.13. Key User Id

The Key User Id attribute indicates the user with whom the key is associated using a distinguished name, as defined in [\[RFC4514\]](#). For example: UID=jsmith,DC=example,DC=net. The attribute definition is as follows:

```
at-pskc-keyUserId ATTRIBUTE ::= {  
    TYPE UTF8String IDENTIFIED BY id-pskc-keyUserId }  
  
id-pskc-keyUserId OBJECT IDENTIFIER ::= { id-pskc 27 }
```

As specified in [\[PSKC\]](#), there are no semantics associated with this element, i.e., there are no checks enforcing that only a specific user can use this key. As such, this element is for informational purposes only.

3.3. Key Policy Attributes

Key policy attributes indicate a policy that can be attached to a key. These attributes are defined in the subsections that follow.

3.3.1. Key Start Date

When included in sKeyAttrs, the Key Start Date attribute indicates the start of the key's validity period. The date MUST be represented in a form that matches the dateTime production in "canonical representation" [[XMLSCHEMA](#)]. Implementations SHOULD NOT rely on time resolution finer than milliseconds and MUST NOT generate time instants that specify leap seconds. The attribute definition is as follows:

```
at-pskc-keyStartDate ATTRIBUTE ::= {  
    TYPE GeneralizedTime IDENTIFIED BY id-pskc-keyStartDate }  
  
id-pskc-keyStartDate OBJECT IDENTIFIER ::= { id-pskc 21 }
```

3.3.2. Key Expiry Date

When included in sKeyAttrs, the Key Expiry Date attribute indicates the end of the key's validity period. The date MUST be represented in a form that matches the dateTime production in "canonical representation" [[XMLSCHEMA](#)]. Implementations SHOULD NOT rely on time resolution finer than milliseconds and MUST NOT generate time instants that specify leap seconds. The attribute definition is as follows:

```
at-pskc-keyExpiryDate ATTRIBUTE ::= {  
    TYPE GeneralizedTime IDENTIFIED BY id-pskc-keyExpiryDate }  
  
id-pskc-keyExpiryDate OBJECT IDENTIFIER ::= { id-pskc 22 }
```

3.3.3. Number of Transactions

The Number of Transactions attribute indicates the maximum number of times a key carried within the package can be used. When this element is omitted there is no restriction regarding the number of times a key can be used. The attribute definition is as follows:

```
at-pskc-noOfTransactions ATTRIBUTE ::= {  
    TYPE INTEGER (0..MAX) IDENTIFIED BY id-pskc-noOfTransactions }  
  
id-pskc-noOfTransactions OBJECT IDENTIFIER ::= { id-pskc 23 }
```


3.3.4. Key Usage

The Key Usage attribute constrains the intended usage of the key. The recipient MUST enforce the key usage. The attribute definition is as follows:

```
at-pskc-keyUsage ATTRIBUTE ::= {  
    TYPE PSKCKeyUsages IDENTIFIED BY id-pskc-keyUsages }  
  
id-pskc-keyUsages OBJECT IDENTIFIER ::= { id-pskc 24 }  
  
PSKCKeyUsages ::= SEQUENCE OF PSKCKeyUsage  
  
PSKCKeyUsage ::= UTF8String ("OTP" | "CR" | "Encrypt" |  
    "Integrity" | "Verify" | "Unlock" | "Decrypt" |  
    "KeyWrap" | "Unwrap" | "Derive" | "Generate")
```

The fields in PSKCKeyUsage have the following meanings:

- o OTP: The key MUST only be used for OTP generation.
- o CR: The key MUST only be used for Challenge/Response purposes.
- o Encrypt: The key MUST only be used for data encryption purposes.
- o Integrity: The key MUST only be used to generate a keyed message digest for data integrity or authentication purposes.
- o Verify: The key MUST only be used to verify a keyed message digest for data integrity or authentication purposes (is converse of Integrity).
- o Unlock: The key MUST only be used for an inverse challenge response in the case where a user has locked the device by entering a wrong PIN too many times (for devices with PIN-input capability).
- o Decrypt: The key MUST only be used for data decryption purposes.
- o KeyWrap: The key MUST only be used for key wrap purposes.
- o Unwrap: The key MUST only be used for key unwrap purposes.
- o Derive: The key MUST only be used with a key derivation function to derive a new key (see also Section 8.2.4 of [[NIST800-57](#)]).

- o Generate: The key MUST only be used to generate a new key based on a random number and the previous value of the key (see also Section 8.1.5.2.1 of [[NIST800-57](#)]).

3.3.5. PIN Policy

The PIN Policy attribute allows policy about the PIN usage to be associated with the key. The attribute definition is as follows:

```
at-pskc-pinPolicy ATTRIBUTE ::= {  
    TYPE PINPolicy IDENTIFIED BY id-pskc-pinPolicy }  
  
id-pskc-pinPolicy OBJECT IDENTIFIER ::= { id-pskc 25 }
```

```
PINPolicy ::= SEQUENCE {  
    pinKeyId          [0] UTF8String OPTIONAL,  
    pinUsageMode      [1] PINUsageMode,  
    maxFailedAttempts [2] INTEGER (0..MAX) OPTIONAL,  
    minLength         [3] INTEGER (0..MAX) OPTIONAL,  
    maxLength         [4] INTEGER (0..MAX) OPTIONAL,  
    pinEncoding       [5] Encoding OPTIONAL }
```

```
PINUsageMode ::= UTF8String ("Local" | "Prepend" | "Append" |  
                             "Algorithmic")
```

The fields in PIN Policy have the following meanings:

- o pinKeyId uniquely identifies the key held within this container that contains the value of the PIN that protects the key.
- o pinUsageMode indicates the way the PIN is used during the usage of the key. The following values are defined in [[PSKC](#)]: Local, Prepend, Append, Algorithmic.
- o maxFailedAttempts indicates the maximum number of times the PIN may be entered wrongly before it MUST NOT be possible to use the key anymore (reasonable values are in the positive integer range of at least 2 and no more than 10).
- o minLength indicates the minimum length of a PIN that can be set to protect the associated key. It MUST NOT be possible to set a PIN shorter than this value. If pinEncoding is 'DECIMAL', 'HEXADECIMAL' or 'ALPHANUMERIC' this value indicates the number of digits/ characters. If pinEncoding is 'BASE64' or 'BINARY', this value indicates the number of bytes of the unencoded value.

- o maxLength indicates the maximum length of a PIN that can be set to protect this key. It MUST NOT be possible to set a PIN longer than this value. If pinEncoding is 'DECIMAL', 'HEXADECIMAL' or 'ALPHANUMERIC' this value indicates the number of digits/characters. If the pinEncoding is 'BASE64' or 'BINARY', this value indicates the number of bytes of the unencoded value.
- o pinEncoding is based on Encoding, which is defined in [Section 3.2.6](#), and specifies encoding of the PIN and MUST be one of the following values: DECIMAL, HEXADECIMAL, ALPHANUMERIC, BASE64, or BINARY.

If pinUsageMode is set to "Local" then the device MUST enforce the restriction indicated in maxFailedAttempts, minLength, maxLength and pinEncoding, otherwise it MUST be enforced on the server side.

[4. Key Encoding](#)

Two parties receiving the same key as an sKey OCTET STRING must make use of the key in exactly the same way in order to interoperate. To ensure that, it is necessary to define a correspondence between the abstract syntax of sKey and the notation in the standard algorithm description that defines how the key is used. The next sections establish that correspondence for the algorithms AES [[FIPS197](#)] and TDEA [[SP800-67](#)].

[4.1. AES Key Encoding](#)

[FIPS197] [section 5.2](#), titled Key Expansion, uses the input key as an array of bytes indexed starting at 0. The first octet of sKey SHALL become the key byte in AES labeled index 0 in [[FIPS197](#)]; the succeeding octets of sKey SHALL become key bytes in AES in increasing index order.

Proper parsing and key load of the contents of sKey for AES SHALL be determined by using the following sKey octet string to generate and match the key expansion test vectors in [[FIPS197](#)] [Appendix A](#) for AES Cipher Key: 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c

Tag Length Value

04 16 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c

[4.2. Triple DES Key Encoding](#)

A Triple-DES key consists of three keys for the cryptographic engine (Key1, Key2, and Key3) that are each 64 bits (56 key bits and 8 parity bits); the three keys are also collectively referred to as a

key bundle [[SP800-67](#)]. A key bundle may employ either two or three independent keys. When only two independent keys are employed (called two-key Triple DES), then the same value is used for Key1 and Key3.

Each key in a Triple-DES key bundle is expanded into a key schedule according to a procedure defined in [[SP800-67](#)] [Appendix A](#). That procedure numbers the bits in the key from 1 to 64, with number 1 being the left-most, or most significant bit (MSB). The first octet of sKey SHALL be bits 1 through 8 of Key1 with bit 1 being the MSB. The second octet of sKey SHALL be bits 9 through 16 of Key1, and so forth, so that the trailing octet of sKey SHALL be bits 57 through 64 of Key3 (or Key2 for two-key Triple DES).

Proper parsing and key load of the contents of sKey for Triple-DES SHALL be determined by using the following sKey octet string to generate and match the key expansion test vectors in [[SP800-67](#)] [appendix B](#) for the key bundle:

Key1 = 0123456789ABCDEF

Key2 = 23456789ABCDEF01

Key3 = 456789ABCDEF0123

Tag Length Value

04 24 0123456789ABCDEF 23456789ABCDEF01 456789ABCDEF0123

[5. Security Considerations](#)

Implementers of this protocol are strongly encouraged to consider generally accepted principles of secure key management when integrating this capability within an overall security architecture.

The symmetric key package contents are not protected. This content type can be combined with a security protocol to protect the contents of the package. One possibility is to include this content type in place of a PSKC package in [[DSKPP](#)] exchanges. In this case, the algorithm requirements are found in those documents. Another possibility is to encapsulate this content type in a CMS [[RFC5652](#)] protecting content type.

[6. IANA Considerations](#)

This document makes use of object identifiers to identify a CMS content type (Appendix A.1), the ASN.1 version of the PSKC attributes (Appendix A.2), and the ASN.1 modules found in [Appendix A.1](#) and A.2.

All OIDs are registered in an arc delegated by RSADSI to the SMIME Working Group. The current contents of the arc are located here:

<http://www.imc.org/ietf-smime/sm-oid.asn>

They were obtained by sending a request to ietf-smime-oid-reg@imc.org. When the SMIME WG closes, this arc and registration procedures will be transferred to IANA. No further action by IANA is necessary for this document or any anticipated updates.

7. References

7.1. Normative References

- [FIPS197] National Institute of Standards. "FIPS Pub 197: Advanced Encryption Standard (AES)", 26 November 2001.
- [IANAPENREG] IANA, "IANA Private Enterprise Number Registry", <<http://www.iana.org/assignments/enterprise-numbers/>>.
- [LUHN] ISO/IEC 7812-1:2006 Identification cards - Identification of issuers - Part 1: Numbering system.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, [RFC 3629](#), November 2003.
- [RFC4049] Housley, R., "BinaryTime: An Alternate Format for Representing Date and Time in ASN.1", [RFC 4049](#), April 2005.
- [RFC4514] Zeilenga, K., "Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names", [RFC 4514](#), June 2006.
- [RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", [RFC 4648](#), October 2006.
- [RFC5646] Phillips, A. and M. Davis, "Tags for Identifying Languages", [BCP 47](#), [RFC 5646](#), September 2009.
- [RFC5911] Hoffman, P., and J. Schaad, "New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME", [RFC 5911](#), June 2010.

- [RFC5912] Hoffman, P., and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", [RFC 5912](#), June 2010.
- [OATHMAN] OATH Manufacturer prefixes:
<http://www.openauthentication.org/oath-id/prefixes/>
- [PSKC] Hoyer, P., Pei, M., and S. Machani, "Portable Symmetric Key Container (PSKC)", [draft-ietf-keyprov-pskc-07.txt](#), work-in-progress.

/** RFC EDITOR: Please replace [[PSKC](#)] with [RFCXXXX] where XXXX is the [draft-ietf-keyprov-pskc](#)'s RFC #. Make the replacements here and elsewhere in the document. **/

- [SP800-67] National Institute of Standards and Technology, "NIST Special Publication 800-67 Version 1.1: Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher", NIST Special Publication 800-67, May 2008.
- [X.680] ITU-T Recommendation X.680 (2002) | ISO/IEC 8824-1:2002. Information Technology - Abstract Syntax Notation One.
- [X.681] ITU-T Recommendation X.681 (2002) | ISO/IEC 8824-2:2002. Information Technology - Abstract Syntax Notation One: Information Object Specification.
- [X.682] ITU-T Recommendation X.682 (2002) | ISO/IEC 8824-3:2002. Information Technology - Abstract Syntax Notation One: Constraint Specification.
- [X.683] ITU-T Recommendation X.683 (2002) | ISO/IEC 8824-4:2002. Information Technology - Abstract Syntax Notation One: Parameterization of ASN.1 Specifications.
- [X.690] ITU-T Recommendation X.690 (2002) | ISO/IEC 8825-1:2002. Information Technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER).
- [XMLSCHEMA] Biron, P., and A. Malhotra, "XML Schema Part 2: Datatypes Second Edition",
<http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>, 2004.

7.2. Informative References

- [DSKPP] Doherty, A., Pei, M., Machani, S., and M. Nystrom, "Dynamic Symmetric Key Provisioning Protocol", Internet Draft Informational, URL: <http://www.ietf.org/internet-drafts/draft-ietf-keyprov-dskpp-10.txt>, work in progress.
- /** RFC EDITOR: Please replace [DSKPP] with [RFCXXXX] where XXXX is the [draft-ietf-keyprov-dskpp](#)'s RFC #. Make the replacements here and elsewhere in the document. **/
- [NIST800-57] National Institute of Standards and Technology, "NIST Special Publication 800-57, Recommendation for Key Management - Part 1: General (Revised)", NIST Special Publication 800-57, March 2007.
- [RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", [RFC 5652](#), September 2009.

APPENDIX A: ASN.1 Module

This appendix provides the normative ASN.1 definitions for the structures described in this specification using ASN.1 as defined in [\[X.680\]](#), [\[X.681\]](#), [\[X.682\]](#), and [\[X.683\]](#).

A.1. Symmetric Key Package ASN.1 Module

```
SymmetricKeyPackageModulev1
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
  smime(16) modules(0) id-mod-symmetricKeyPkgV1(33) }

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

-- EXPORTS ALL

IMPORTS

-- From New PKIX ASN.1 [RFC5912]

ATTRIBUTE
FROM PKIX-CommonTypes-2009
{ iso(1) identified-organization(3) dod(6) internet(1)
  security(5) mechanisms(5) pkix(7) id-mod(0)
  id-mod-pkixCommon-02(57) }
```


-- From New SMIME ASN.1 [[RFC5911](#)]

```
CONTENT-TYPE, Attribute{}
  FROM CryptographicMessageSyntax-2009
    { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
      smime(16) modules(0) id-mod-cms-2004-02(41) }

;

ContentSet CONTENT-TYPE ::= {
  ct-symmetric-key-package,
  ... -- Expect additional content types --
}

ct-symmetric-key-package CONTENT-TYPE ::=
  { SymmetricKeyPackage IDENTIFIED BY id-ct-KP-sKeyPackage }

id-ct-KP-sKeyPackage OBJECT IDENTIFIER ::=
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
    smime(16) ct(1) 25 }

SymmetricKeyPackage ::= SEQUENCE {
  version          KeyPkgVersion DEFAULT v1,
  sKeyPkgAttrs    [0] SEQUENCE SIZE (1..MAX) OF Attribute
                                     {{ SKeyPkgAttributes }} OPTIONAL,
  sKeys           SymmetricKeys,
  ... }

SymmetricKeys ::= SEQUENCE SIZE (1..MAX) OF OneSymmetricKey

OneSymmetricKey ::= SEQUENCE {
  sKeyAttrs    SEQUENCE SIZE (1..MAX) OF Attribute
                                     {{ SKeyAttributes }} OPTIONAL,
  sKey        OCTET STRING OPTIONAL }
  ( WITH COMPONENTS { ..., sKeyAttrs PRESENT } |
    WITH COMPONENTS { ..., sKey PRESENT } )

KeyPkgVersion ::= INTEGER { v1(1) } ( v1, ... )

SKeyPkgAttributes ATTRIBUTE ::= { ... }

SKeyAttributes ATTRIBUTE ::= { ... }

END
```


A.2. PSKC ASN.1 Module

```
PSKCAAttributesModule
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
    smime(16) modules(0) id-mod-pskcAttributesModule(53) }

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

-- EXPORTS ALL

IMPORTS

-- From New PKIX ASN.1 [RFC5912]

ATTRIBUTE
  FROM PKIX-CommonTypes-2009
  { iso(1) identified-organization(3) dod(6) internet(1)
    security(5) mechanisms(5) pkix(7) id-mod(0)
    id-mod-pkixCommon-02(57) }

-- From BinaryTime [RFC4049]

BinaryTime
  FROM BinarySigningTimeModule
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
    smime(16) modules(0) id-mod-binarySigningTime(27) }

-- From New SMIME ASN.1 [RFC5911]

id-smime
  FROM SecureMimeMessageV3dot1-2009
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
    smime(16) modules(0) id-mod-msg-v3dot1-02(39) }

;

--
-- PSKC Attributes OIDs are taken from the SMIME Arc.
--

id-pskc OBJECT IDENTIFIER ::= { id-smime 12 }
```

```
--
-- Merge SKeyPKGAttributes to the set of attributes for sKeyPkgAttrs
--

SKeyPkgAttributes ATTRIBUTE ::= {
    at-pskc-manufacturer | at-pskc-serialNo | at-pskc-model |
    at-pskc-issueNo | at-pskc-deviceBinding |
    at-pskc-deviceStartDate | at-pskc-deviceExpiryDate |
    at-pskc-moduleId | at-deviceUserId, ... }

--
-- Merge SKeyAttributes to the set of attributes for sKeyAttrs
--

SKeyAttributes ATTRIBUTE ::= {
    at-pskc-keyId | at-pskc-algorithm | at-pskc-issuer |
    at-pskc-keyProfileId | at-pskc-keyReference |
    at-pskc-friendlyName | at-pskc-algorithmParameters |
    at-pskc-counter | at-pskc-time | at-pskc-timeInterval |
    at-pskc-timeDrift | at-pskc-valueMAC | at-keyUserId |
    at-pskc-keyStartDate | at-pskc-keyExpiryDate |
    at-pskc-numberOfTransactions | at-pskc-keyUsage |
    at-pskc-pinPolicy,... }

at-pskc-manufacturer ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-manufacturer }

id-pskc-manufacturer OBJECT IDENTIFIER ::= { id-pskc 1 }

at-pskc-serialNo ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-serialNo }

id-pskc-serialNo OBJECT IDENTIFIER ::= { id-pskc 2 }

at-pskc-model ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-model }

id-pskc-model OBJECT IDENTIFIER ::= { id-pskc 3 }

at-pskc-issueNo ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-issueNo }

id-pskc-issueNo OBJECT IDENTIFIER ::= { id-pskc 4 }

at-pskc-deviceBinding ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-deviceBinding }
```



```
id-pskc-deviceBinding OBJECT IDENTIFIER ::= { id-pskc 5 }

at-pskc-deviceStartDate ATTRIBUTE ::= {
    TYPE GeneralizedTime IDENTIFIED BY id-pskc-deviceStartDate }

id-pskc-deviceStartDate OBJECT IDENTIFIER ::= { id-pskc 6 }

at-pskc-deviceExpiryDate ATTRIBUTE ::= {
    TYPE GeneralizedTime IDENTIFIED BY id-pskc-deviceExpiryDate }

id-pskc-deviceExpiryDate OBJECT IDENTIFIER ::= { id-pskc 7 }

at-pskc-moduleId ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-moduleId }

id-pskc-moduleId OBJECT IDENTIFIER ::= { id-pskc 8 }

at-pskc-deviceUserId ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-deviceUserId }

id-pskc-deviceUserId OBJECT IDENTIFIER ::= { id-pskc 26 }

at-pskc-keyId ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-keyId }

id-pskc-keyId OBJECT IDENTIFIER ::= { id-pskc 9 }

at-pskc-algorithm ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-algorithm }

id-pskc-algorithm OBJECT IDENTIFIER ::= { id-pskc 10 }

at-pskc-issuer ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-issuer }

id-pskc-issuer OBJECT IDENTIFIER ::= { id-pskc 11 }

at-pskc-keyProfileId ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-keyProfileId }

id-pskc-keyProfileId OBJECT IDENTIFIER ::= { id-pskc 12 }

at-pskc-keyReference ATTRIBUTE ::= {
    TYPE UTF8String IDENTIFIED BY id-pskc-keyReference }

id-pskc-keyReference OBJECT IDENTIFIER ::= { id-pskc 13 }
```



```
at-pskc-friendlyName ATTRIBUTE ::= {
  TYPE FriendlyName IDENTIFIED BY id-pskc-friendlyName }

id-pskc-friendlyName OBJECT IDENTIFIER ::= { id-pskc 14 }

FriendlyName ::= SEQUENCE {
  friendlyName      UTF8String,
  friendlyNameLangTag UTF8String OPTIONAL }

at-pskc-algorithmParameters ATTRIBUTE ::= {
  TYPE PSKCAAlgorithmParameters
  IDENTIFIED BY id-pskc-algorithmParameters }

id-pskc-algorithmParameters OBJECT IDENTIFIER ::= { id-pskc 15 }

PSKCAAlgorithmParameters ::= CHOICE {
  suite          UTF8String OPTIONAL,
  challengeFormat [0] ChallengeFormat,
  responseFormat [1] ResponseFormat,
  ... }

ChallengeFormat ::= SEQUENCE {
  encoding      Encoding,
  checkDigit    BOOLEAN DEFAULT FALSE,
  min           INTEGER (0..MAX),
  max           INTEGER (0..MAX),
  ... }

Encoding ::= UTF8String ("DECIMAL" | "HEXADECIMAL" |
  "ALPHANUMERIC" | "BASE64" | "BINARY" )

ResponseFormat ::= SEQUENCE {
  encoding      Encoding,
  length        INTEGER (0..MAX),
  checkDigit    BOOLEAN DEFAULT FALSE,
  ... }

at-pskc-counter ATTRIBUTE ::= {
  TYPE INTEGER (0..MAX) IDENTIFIED BY id-pskc-counter }

id-pskc-counter OBJECT IDENTIFIER ::= { id-pskc 16 }

at-pskc-time ATTRIBUTE ::= {
  TYPE BinaryTime IDENTIFIED BY id-pskc-time }

id-pskc-time OBJECT IDENTIFIER ::= { id-pskc 17 }
```



```
at-pskc-timeInterval ATTRIBUTE ::= {
  TYPE INTEGER (0..MAX) IDENTIFIED BY id-pskc-timeInterval }

id-pskc-timeInterval OBJECT IDENTIFIER ::= { id-pskc 18 }

at-pskc-timeDrift ATTRIBUTE ::= {
  TYPE INTEGER (0..MAX) IDENTIFIED BY id-pskc-timeDrift }

id-pskc-timeDrift OBJECT IDENTIFIER ::= { id-pskc 19 }

at-pskc-valueMAC ATTRIBUTE ::= {
  TYPE ValueMac IDENTIFIED BY id-pskc-valueMAC }

id-pskc-valueMAC OBJECT IDENTIFIER ::= { id-pskc 20 }

ValueMac ::= SEQUENCE {
  macAlgorithm UTF8String,
  mac          UTF8String }

at-pskc-keyUserId ATTRIBUTE ::= {
  TYPE UTF8String IDENTIFIED BY id-pskc-keyUserId }

id-pskc-keyUserId OBJECT IDENTIFIER ::= { id-pskc 27 }

at-pskc-keyStartDate ATTRIBUTE ::= {
  TYPE GeneralizedTime IDENTIFIED BY id-pskc-keyStartDate }

id-pskc-keyStartDate OBJECT IDENTIFIER ::= { id-pskc 21 }

at-pskc-keyExpiryDate ATTRIBUTE ::= {
  TYPE GeneralizedTime IDENTIFIED BY id-pskc-keyExpiryDate }

id-pskc-keyExpiryDate OBJECT IDENTIFIER ::= { id-pskc 22 }

at-pskc-numberOfTransactions ATTRIBUTE ::= {
  TYPE INTEGER (0..MAX) IDENTIFIED BY id-pskc-numberOfTransactions }

id-pskc-numberOfTransactions OBJECT IDENTIFIER ::= { id-pskc 23 }

at-pskc-keyUsage ATTRIBUTE ::= {
  TYPE PSKCKeysUsages IDENTIFIED BY id-pskc-keyUsages }

id-pskc-keyUsages OBJECT IDENTIFIER ::= { id-pskc 24 }

PSKCKeysUsages ::= SEQUENCE OF PSKCKeysUsage
```

```
PSCKKeyUsage ::= UTF8String ("OTP" | "CR" | "Encrypt" |  
    "Integrity" | "Verify" | "Unlock" | "Decrypt" |  
    "KeyWrap" | "Unwrap" | "Derive" | "Generate")
```

```
at-pskc-pinPolicy ATTRIBUTE ::= {  
    TYPE PINPolicy IDENTIFIED BY id-pskc-pinPolicy }
```

```
id-pskc-pinPolicy OBJECT IDENTIFIER ::= { id-pskc 25 }
```

```
PINPolicy ::= SEQUENCE {  
    pinKeyId          [0] UTF8String OPTIONAL,  
    pinUsageMode      [1] PINUsageMode,  
    maxFailedAttempts [2] INTEGER (0..MAX) OPTIONAL,  
    minLength         [3] INTEGER (0..MAX) OPTIONAL,  
    maxLength         [4] INTEGER (0..MAX) OPTIONAL,  
    pinEncoding       [5] Encoding OPTIONAL }
```

```
PINUsageMode ::= UTF8String ("Local" | "Prepend" | "Append" |  
    "Algorithmic")
```

END

Authors' Addresses

Sean Turner
IECA, Inc.
3057 Nutley Street, Suite 106
Fairfax, VA 22031
USA

EMail: turners@ieca.com

Russ Housley
Vigil Security, LLC
918 Spring Knoll Drive
Herndon, VA 20170
USA

EMail: housley@vigilsec.com